A unified modeling framework for service design

Jayasinghe Arachchig, J.

Publication date:
2013

Document Version
Publisher's PDF, also known as Version of record

A Unified Modeling Framework for Service Design

Jeewanie Jayasinghe Arachchige

March 15, 2013

A Unified Modeling Framework for Service Design

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan Tilburg University, op gezag van de rector magnificus, prof. dr. Ph. Eijlander, in het openbaar te verdedigen ten overstaan van een door het college voor promoties aangewezen commissie in de aula van de Universiteit op vrijdag 15 maart 2013 om 10.15 uur

door

Jeewanie Jayasinghe Arachchige

geboren op 25 juni 1972 te Matara, Sri Lanka

PROMOTIECOMMISSIE:

PROMOTOR:	prof. dr. P.M.A.Ribbers
COPROMOTORES:	dr. H. Weigand
	dr. M. A. Jeusfeld
OVERIGE LEDEN:	prof.dr. ir. H.A.M. Daniels
	dr. M.V. Dignum
	dr. J. Verelst
	dr. P. Hruby

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems (Dissertation Series No. 2013-10), and CentER, the Graduate School of the Faculty of Economics and Business Administration of Tilburg University.

Cover Design: dr. J.N. Handunge

ආදරණීය මෑණියන්ට සහ පියාණන්ට!

To my loving mother and father

Acknowledgement

This dissertation would not have been possible without the guidance and the help of many individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study. I am trying to mention few of them in this page.

First of all, I would like to offer very special thank to my supervisor dr. Hans Weigand for his excellent assistance received during my research work. His understandability, patience, motivation and immense knowledge help me to achieve my goal. He is such a kind-hearted person I have ever met.

Then I wish to pay my gratitude to dr. Manfred Jeusfeld, a core- promoter for his valuable support specially when converting the theory into workable solution using the tool ConceptBase.

Special thank goes to prof. dr. Piet Ribbers for being the Promoter of this research study.

Then, I am thankful to our secretaries Alice Kloosterhuis and Mieke Smulders for their all the support during the period in Tilburg University.

The next person which I am very grateful is dr. Prasad Jayaweera- a senior lecturer, Jayawardhanapura University, Sri Lanka. Without him, I would not be able to obtain a PhD placement in Tilburg University. In early stages of my carrier as a lecturer at University of Ruhuna, he is the one who motivated me to step into the research world. He referred me to dr. Hans Weigand as a potential PhD candidate.

I acknowledge the several personnels of University of Ruhuna, especially dr. P.A. Jayantha (former dean of the Faculty of Science), Mr. S. A. S Lorensuhewa (former head of the department - Computer Science) and dr.D.B Guruge (head of the department - Computer Science) for their assistance.

I am very grateful to my friends in the department to who stand behind me in different circumstances and who share the life in Tilburg University. Even some of them are already left the department I would like to mention names of Williem, Christina, Marcel, and Jon. Then Khoa and our group members Patricio and Irfan come to the list. My special thanks go to my good girl friends Faiza and Yan. Faiza and me as office mates use to through away our common problems and feelings into the air in room no. K718. Yan is always with me when I need a help. Thank you very much for everything.

Then, I mention the people who are always very closer to my heart. First among them, my heart-filled thank goes my dear mother and father for being my inner spirit at every moment of my life. Then I write a sentence for my two sisters (Krishanthi and Nesha) and cousin sister (Bilani) who always love me and with me whatever happen to me.

Finally, I would not be able to write a single word of this dissertation without my loving husband, Jagath. Being a part and becoming the biggest strength of my life, he shared and bared everything with me.

At last but not least, these words for our ever loving little son- Saunak, the most precious gift I got during my PhD. Every hard time, every tired day and unbearable pressure, are vanished with a single smile of his little face.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Constantly changing customer needs create new opportunities and challenges to the businesses in the world. According to Baida et al. (2004), customers do not buy goods or services: they buy offerings which render services which create value. Gronroos (2000) pointed out the same idea. According to him, the physical goods become one element among others in a total service offering. These statements are further strengthened by looking at the economical statistical figures. In 2011, services comprise roughly 80 percent of economic activities in the United States, and more than 60 percent of economic activities in the top economies in the world (OECD, 2012). Therefore, it is clear *Service* is becoming front and center abstraction not only in marketing and the economics perspective but also in customer's perspective. The growth of the service-centered economy changes the nature of businesses in the world. Gronroos (2000) says that physical goods marketing and services marketing converge, but services-oriented thinking will dominate. Currently, the service-oriented thinking has become not only the de facto standard but also an integral part of the business itself (Nayak et al., 2006).

1.1 Background

As service means different thing to different people, let's start by defining what is meant by *service.* Baida et al. (2004) discussed the notion of service from three perspectives namely service in economics and business, service in computer science and service in information technology. Among many definitions, we selected some comprehensive ones starting from business perspective, then to the software perspective and lastly a general definition to both. Finally, we provide the meaning of the word *service*, which is used in this thesis.

Service definitions with a business perspective:

> *"An act or performance that one party can offer to another that is essentially intangible and does not result in the ownership of anything",* Kotler and Keller (2006).

> "*A change in the condition of a person, or a good belonging to a some economic entity, brought about as the result of the activity of some other economic entity, with approval of first person economic entity*", Hill (1977).

Service definitions with a software perspective:

> *"A service is generally implemented as a course-grained, discoverable software entity that exists as a single instance and interacts with applications and other services through a loosely coupled, message based communication model",* Brown et al. (2005).
>
> *"loosely coupled, reusable software components that semantically encapsulate discrete functionality and are distributed and programmatically accessible over standard Internet protocols",* The Stencil Group (2001).

General Service definition:

> *"Services are acts performed for other entities including the provision of resources that other entities will use*, Alter (2012).

In this research we use the term "service" to represent "*an activity which is offered by one party to another that brings a desired change*". If we refer to software services, we use the prefix (such as software or web) in the rest of this thesis.

Service orientation in a business intends that the entire business including, business strategy, customer relations, supplier relations, IT infrastructure, employees, etc. are oriented towards providing service. This situation changes the design, implementation and deploying of enterprise information systems as well.

Enterprises gain many advantages when framing the business using Service-Oriented Architectures (SOA). SOA is considered as an approach that enables the allocation of business activities among business partners in a value chain (Chesbrough and Spohrer, 2006). Nayak et al. (2006), Gronroos (2000) also highlight the popularity of SOA with business point of view.

The following researchers address the usage of SOA from a more technical perspective. Erl (2005) says "SOA as an architectural style has been widely adopted in industry thanks to its ability of providing seamless integration among software services". Papazoglou (2007) mentioned that SOA provides major advantages for today's enterprise information systems by presenting the interfaces that loosely coupled connections require.

Having realized the advantages of SOA, many researchers (Holley et al., 2003; Alter, 2012; De Castro et al., 2009; Arsanjani et al., 2008) have started to introduce service-oriented thinking in software application design and implementation. Although the SOA has been under discussion for several years, it is mainly regarded as a technical concept for the integration of heterogeneous application environments (Kohlmann and Alt, 2009). Most research on this topic so far takes a strict software engineering perspective. However, as stated in (Nayak et al., 2007), "the current trend toward a service-oriented enterprise necessitates a formal characterization of business architecture that reflects service-oriented business thinking". In order to cope with these deficiencies, several researchers integrate business thinking in the services design. For instance, Weigand et al., (2009);

Zimmerman et al., (2004); Henkel et al., (2007); Alter, (2012); Arsanjani et al., (2008); Haesen, (2009).

Weigand et al. (2009) build on established business ontologies (REA and e3-value) to develop a value-based service design method. Following a top-down approach the authors pay attention to identifying business services as an input to the web service identification. This framework suggests classifying the business services and applicable policies into a tabular form. Service modeling as such is not the scope of the above research work.

The Service-Oriented Modeling and Architecture (SOMA) methodology of IBM (Arsanjani et al., 2008) is another approach which minimizes the gap between business and ICT. Although SOMA supports incorporate business thinking using several methods and model such as business use cases, goal-oriented service modeling to their method, it does not support service modeling at the business level.

The service modeling techniques listed above clearly show the need of business service modeling for proper alignment between the business and the IT. (The detailed discussion of related work is available in chapter 2). The starting point for design should be the business level at which services can be identified that provide value to customers and can be offered in an economically viable way. It is at the business level that business concerns should be dealt with first.

In this dissertation, we propose a business service modeling framework to capture the real business activities as services. As the Model Driven Architecture (MDA) provides a useful framework for the software development, we adopt some MDA concepts for the proposed service modeling framework. The Model-Driven Architecture provides a framework for software development that uses models to describe the system to be built. These models can be expressed at various levels of abstraction, with each level emphasizing certain aspects or viewpoints of the system (Mellor et al., 2002). One of the main advantages of MDA approach is the definition of a conceptual structure where the models used by business managers and analysts can be traced towards more detailed models used by software developers (De Castro et.al, 2007). In MDA, the high-level business view is represented by means of Computational Independent Models (CIM) while the information system view is represented using Platform Independent Models (PIM) and Platform Specific Models (PSM). In our work, the business service models are positioned at the CIM level. Another important feature of MDA is transformations between different levels of models. Typically MDA defines transformation rules between PIM and PSM models but just traceability relations between business requirement of the CIM models and the elements of the PIM and PSM models (De Castro et.al, 2007). Miller and Mukerji (2003) state that, "the mapping description may be in natural language, an algorithm in an action language or a model in a mapping language". Following these ideas, we propose a mapping description between CIM which is business service models and PIM levels BPMN- process models only. The mapping is presented using natural language. The transformation between PIM and PSM towards to code generation is not in the scope of this research.

One major pillar of the framework is business service and resource modeling language called BSRM, which is based on the well-known business ontology (Resource-Event-Agent). The next pillar of the framework is business service patterns which are presented using BSRM language. The pattern structure specifications and their operations are defined in a formal way using Bottoni et al. (2010) as a basis. The patterns facilitate to grasp the domain concepts easily. The business service patterns together with design steps provide comprehensive tool set to derive business service model for an enterprise. The usability of the business service model is further strengthened by proposing an integration method of these patterns in web service discovery. We validate the proposed framework with several validation methods including three empirical case studies.

This chapter is structured as follows: Section 1.2 describes the motivation of this research. Then we move to the research goals in section 1.3. The research questions are explained in section 1.4. Section 1.5 describes the research methodology that is applied in this research. The contributions of this research are discussed in section 1.6. The limitations of this research work are discussed in section 1.7. The chapter ends with describing the structure of the whole dissertation.

1.2 Research Motivation

As service-oriented thinking is gaining increasing popularity in today's enterprise information systems, several studies and researches (eg; Holley et al., 2003; Alter, 2012; De Castro et al., 2009; Arsanjani, et al., 2008) have started to establish the service-oriented thinking in software application design and implementation. On the other hand, some researches highlight that the progression of service-oriented thinking is still unable to address real business needs. For example, Alter (2012) states "while the technical architecture has many advantages related to computer systems and networks, the underlying logic of interactions does not translate well to services provided by one person to another". Cummins (2010) says, "the full potential of SOA is realized when it is applied as an architecture for business design". The authors of the following studies also argue that considerable attempt in service-oriented design is still focusing on the operational or software engineering level (Weigand et al., 2009; Zimmerman et al., 2004; Barney,1991; Curdera et al., 2003). However, the authors of (Nayak et al., 2007) point out that the current trend towards a service-oriented enterprise necessitates a formal characterization of business architecture that reflects service-oriented business thinking.

To deal with business aspects when designing information systems, there exist a number of approaches which are able to grasp the basic concepts and their relationships of a business. For example, e^3-value network, Resource-Event-Agent (REA) ontology. For any kind of commercial business the value creation has always been one of the most important business concerns. The term value spans number of disciplines including economics, sociology, anthropology, psychology and marketing. With economics and business perspective Graeber (2001) discussed the definition of value as a person's willingness to pay the price of a good in terms of cash in return for certain product benefits, or experience. The definition highlights the quantitative aspect of value, which is of course quite important from a business perspective. However, it does not say anything about the subjective value experience nor about the internal structure of the value object. Recently, business researchers have started to recognize the intimate relationship between value creation and service provisioning. Vargo and colleagues (Vargo and Lusch, 2004; Alter, 2008) have introduced the concept of "service-dominant (S-D) logic". S-D logic focuses on service provision in contrast to goods production (G-D logic). The shift from G-D to S-D logic is one from a value proposition consisting of operand (passive) resources to one consisting of operant (active) resources. Instead of only seeing value being created within companies that exchange the means for this value creation from one to another, it emphasizes the value being created between companies (or companies and consumers). Gronroos (2000) goes even further by saying that value is primarily created at and by the customer, and the company is just co-creator. If value creation is the main business concern, neither software models nor business processes provide the right level of abstraction for business modeling.

Having realized the importance of service design with business perspective, several researchers have tried to start the design with business perspectives (e.g. Weigand et al., 2009; Zimmerman et al., 2004; Henkel et al., 2007; Alter, 2012; Kohlmann and Alt, 2009, etc). However, in our view these approaches do not go far enough. For example, Zimmerman et al. (2004) distinguish between top-down, bottom-up and meet-in-the-middle approaches and discuss major principles of service design such as low coupling and high cohesion. Although these criteria are relevant, the approach considers service design mainly as a software development problem. IBM's SOMA Arsanjani, et al. (2008) which is a software development life-cycle method for designing and building SOA-based solutions also uses a meet-in-the-middle approach. SOMA links the system level to strategic business goals, but it does not support service modeling at the business level. Alter (2012) proposes guidelines to form a genuinely service oriented enterprise, using work system theory (Alter, 2006). He views the business services and software services as in the same line. But in our view, these two cannot be treated in the same way, as they exist in two different domains. Even though the above research works attempt to bridge the gap between service-oriented thinking at technical level and business level, the service representation at business level is lacking. Even though, some of above approaches (eg: Weigand et al., 2009; De Castro et al. 2009) incorporate existing business modeling techniques to the CIM level, designing services at business level is not addressed. Therefore, a truly service-oriented design mechanism at the business level, where the real services are positioned, is still missing.

1.3 Research Goals

As argued in section 1.1, a service perspective at the business level is an essential requirement for a truly service-oriented design. Both the system designers and the business users can benefit, as a result of applying a service perspective at business level. The main outcome of this research work is a unified framework for service design in terms of patterns based business service modeling. *The main goal is to capture the business activities as services.* As the service design framework intends to capture the business logic with service perspective, the constructs of the framework are rooted in a well-established business ontology- REA. We discuss the research goals from two perspectives: designer's perspective and business perspective. The research goals are described as below.

1. From the designer's perspective, the framework must provide a methodological way to build and maintain the service model. The sub-goals of the above statement are given below.

    - The constructs with their relationships and the design steps should be provided by the framework.
    - As the business environment is volatile, the framework must provide a methodological support to capture new requirements.
    - The framework must provide the best practices in the business with service perspective.
    - The framework must provide an easy way to start design the constructs without starting from the scratch. This helps to reduce the designer's time.
    - The framework must enhance the visibility of services throughout the model and as well as the given case. (in terms of definitions of services, categories of services and their role)
    - As it is not possible to represent all the aspects in one model, the framework must support to trace the service model to other models such as a process model.
    - The framework must support the integration of the service model with web service discovery.

2. From business perspective, the framework must provide a service-driven view to the business. Sub-goals of the above statement are given below.

    - The service models derived from the framework must be easy to understand by non-technical users.
    - The framework must support the translation of business requirements into a service model.
    - The service models derived from the framework should cover all kinds of services and all business layers (for instance, operational layer and management layer).
    - The framework must support any business (manufacturing, service and trading domains) help to become more service oriented.

1.4 Research Questions

Service-oriented architectures are the upcoming business standard for realizing enterprise information systems. As stated in the section 1.1 and 1.2, most research on this topic so far takes a software engineering perspective. The need of the service design with business perspective is emphasized in section 1.2. A truly service-oriented design mechanism at the business level is still missing. In order to find a solution to the above need, we came up with the following research question.

> *"How can service-oriented thinking at business level be materialised in information system design?"*

The research question is further decomposed into several sub questions.

1. *What is the State of the Art in service-oriented design which incorporates business thinking and does it really address the business perspective?*

2. *What is the business model/ontology underlying the business services?*

3. *How is the proposed service modeling language represented?*

4. *How does the proposed framework support the designer building enterprise model to view the business activities in the entire enterprise as services?*

5. *How does the proposed framework support the designer to capture the best practices in the business as services?*

6. *Is the proposed framework flexible enough to capture new business requirements?*

7. *How does the proposed framework support to synchronise the business service model with other models?*

8. *Is the proposed service design framework truly service-oriented?*

9. *How is the proposed service design framework validated in terms of completeness and correctness?*

1.5 Research Methodology

Based on Hevner et al. (2004), the research conducted in this dissertation follows the *design science* approach in information systems. The artifact that we are going to achieve as outcome of in this research is a unified service design framework. The research methodology consists of four main phases namely *problem definition*, *analysis of State of the Art*, *solution design* and *validation*.

Problem definition:
Defining the problem is the primary step of solving a problem. Based on the literature survey and the motivation of the research, we set the goals of this research. In order to achieve the research goals, the research problem has to be defined. Hence, we define our research problem in section 1.3 as research questions.

Analysis of State of the Art:
Investigating the existing literature clarifies the research question and goals. A comprehensive literature survey presented in chapter 2 gives opportunity to identify new research challenges and reusability of previous knowledge. A feature comparison of selected existing approaches is done in chapter 2. The analysis of State of the Art helps to improve the new solution design.

Solution design:
Having realized the shortcomings and the strengths of existing approaches of the service-oriented design at business level, we design a new solution. Finding the right approach for solving a world problem is a *knowledge problem* (Wieringa et al. 2006). Knowledge problems represent a lack of knowledge about the world. In this research, we use knowledge from business modeling techniques and SOA design. The solution that we design is a unified service design framework. The results of this research are described in Chapter 3, 4 and 5.

Validation:
The validation of the solution is performed in two ways. In the first place, we validate the results of this research using non-empirical formal approaches (e.g., feature comparison, metamodeling). Secondly, the proposed framework is applied to three cases selected from manufacturing and service domain. These results are discussed in chapter 6.

1.6 Contributions

This dissertation addresses the need for business thinking in service-oriented design and proposes a framework for truly service-oriented design. The main contribution of this research is the service design framework. The framework consists of a business service modeling language called BSRM (business service and resource modeling) and a business service pattern library. The methods proposed in the literature that incorporate existing business modeling techniques for service-oriented design are not able to provide a language to model the business activities as service.

The constructs of the language are defined using a metamodel which is grounded in a well-known business ontology – REA. The modeling language enables the designer to grasp the business concepts with service view. BSRM gives a better insight into the value co-creation which is the main focus of services. The BSRM language has been published in (Jayasinghe et al., 2012). It is further explained in chapter 3.

The research also proposes a classification of business services. The separation of services through classification has many advantages. For example, the operational activities and supporting activities can be modeled separately. These supporting activities can be improved further, as they modeled separately. In chapter 5, under enhance services, we discuss the supporting services in detail. We published one category of enhance service (Weigand et al., 2011).

The next main outcome of this research is a business service pattern library together with a comprehensive pattern composition mechanism. Pattern based approaches are common in software engineering. In particular, Hruby describes more than 20 business patterns founded on REA in his book (Hruby, 2006). (more details are available in chapter 2). As he follows MDA and approach and REA basis for defining business patterns, we selected same examples from his work wherever possible. The novelty of our approach is, pattern based model completion and integration to web service discovery. We follow the formal approach proposed by Bottoni et al. (2010) for pattern specification and pattern composition. The business service patterns are defined using pattern structure specifications. We propose a pattern operator called*" Merge"* for proper composition of patterns. The patterns, pattern operators together with design steps provide a comprehensive approach to pattern composition to build the service model. Business service patterns are defined to represent the basic activities in the manufacturing domain. The role of business service patterns is not limited to the design of business services, but does also support service integration at implementation level. Using the service integration metamodel, the changes in the business services are aligned with corresponding changes in the business processes. The results were published in (Jayasinghe and Weigand, 2012a) and (Jayasinghe and Weigand, 2012b). Chapter 4 discusses the business service patterns.

1.7 Limitations

The limitations of the proposed service design framework are discussed below.

- The validation is limited to three case studies in literature. One case is a fictional case about Global Bike Inc., a bike manufacturing company presented by SAP (Magel and Word, 2012). The second one is a real world case about wine production presented by S-Cube (S-Cube, 2009). Both cases relate to manufacturing domain. The next case is from logistic domain. It describes a transportation service provided by a Dutch Transport company. The validation of the framework is limited to the above domains. The proposed framework has not been validated in a field experiment.

- The service modeling framework consists of business service patterns. The business service patterns represent the selected activities in manufacturing domain. The business service pattern library can be extended further within the manufacturing domain and also to other domains. Our claim is that the proposed framework can be applied to any kind of business. However, we didn't investigate where the framework is most suitable.

- We are not claiming that the proposed modeling notation is complete. The notation is sufficient to represent the basic construct of the BSRM language. It doesn't support modeling the constraints. The modeling notation is also lacking the representation of constraints.

1.8 Structure of the Dissertation

The structure of the thesis is as follows.

Chapter 2:
Chapter two sets the background of this research and discusses the related work in service-oriented design with business perspective. Under the background, we explain the business models and how business models are used in SOA. Then we discuss the business modeling ontologies and the motivation of selecting the REA business ontology as the basis of our service metamodel. The first part of the chapter ends with

a overview of business patterns. The second part of the chapter is dedicated to the related work in service-oriented design with business perspective.

Chapter 3:
Chapter three is the basis of the service modeling framework. The service modeling language called Business Service and Resource Model (BSRM) is described in chapter 3. The constructs of the language are based on a metamodel. Therefore section 3.1, describes the metamodel for the service modeling language. It includes BSRM Language Semantics. In section 3.2, we describe the implementation of the metamodel with a modeling tool called ConceptBase. Then we move to the modeling notation and the service design examples in sections 3.3 and 3.4 accordingly. The traceability of the BSRM model with other models is discussed in subsection 3.5. The section 3.6 - BSRM Viability Check contains, among others, a comparison of selected features of BSRM with other models. Finally, this chapter ends with a section called Service Model Analysis (section 3.7) which gives a general overview of value analysis with BSRM.

Chapter 4:
Chapter 4 describes the Business Service Patterns (BSP). The pattern structure specification is described in section 4.1. Five pattern structure specifications are explained under this section. The pattern operations (expansion and annotation) on pattern structures are given in section 4.2. We discuss the business service alignment with value activities in section 4.3. Then we demonstrate generic business service patterns for primary and supporting activities in section 4.4 and section 4.5 respectively. Several generic business service patterns for service industry are described in section 4.6. The pattern composition with business pattern operators is defined in section 4.7. We describe the design steps to compose the enterprise model, in section 4.8. The chapter ends with describing the integration of business services with software services.

Chapter 5:
Chapter 5 explains a special category of services namely *enhance* service. This chapter has four sub sections to describe the four categories of the enhance services namely *management* as a service, *human resource provisioning* as a service, *publication* as a service and *access* as a service.

Chapter 6:
Chapter 6 presents the empirical validation of this framework using three case studies. First case study, which is about Global Bike Inc. (Magel and Word, 2012), a fictional case by SAP, is given in section 6.2. The second case study about wine production from S-Cube (S-Cube, 2009) is illustrated in section 6.3. The third case is about Transportation Company (Dieleman, 2010).

Chapter 7:
The dissertation is wrapped up with the conclusion of this research and the future work.

Chapter 2

Background and Related Work

In this chapter, we establish the background of our work and discuss related work in service design. The first part of the chapter (section 2.1~ section 2.4) is about the background and the rest of the chapter is dedicated to the related work. As described in the motivation, the main focus of this research is establishing a service-oriented design mechanism that is starting from the business level. Business thinking in service design plays an important role in this research. Therefore, we start the chapter with business modeling. The next section is about reference models. Then we move on describing business modeling ontologies and selecting one ontology as a basis of our work. The next section describes the Resource, Event and Agent (REA) ontology. As the business service patterns are an important pillar of the proposed service modeling framework, we include a discussion about business patterns in this chapter. Moving on, we present the related work of service oriented design in the second part of this chapter. The chapter ends with a comparison of different approaches in service design.

2.1 Business Modeling

Different business models address different aspects of the business. One of the most cited definitions for business model is given by Timmer (1998). He defines the business model as "an architecture for the product, service and information flows, including a description of the various business actors and their roles; and a description of the potential benefits for the various actors; and description of the sources of revenues". The authors of (Osterwalder and Pigneur, 2002) classified these aspects into three main categories namely revenue- and product specific, business actor- and network-specific and marketing-specific. There are several advantages of using business models. According to the (Osterwalder and Pigneur, 2002) and (Gordijn and Akkermans, 2001), business models help to understand the key elements and mechanisms in a specific business domain and their relationships. The authors of (Eriksson and Penker, 2000) say that the business models help to specify valid requirements for the Information Systems. The authors of Scheithauer et al. (2009) points out that business modeling is a discipline which depicts the relationship between business logic and its realization with information technology. The authors of Bloch et al. (1996); Kalakota and Robinson (1999); Maître and Aladjidi (1999) state that new business models are constantly emerging in electronic commerce and can become a major stake in the e-business. The researchers of Hruby (2006); Gordijn et al. (2000) ; Andersson et al. (2005) also suggest that when addressing business-IT integration, constructive results are obtained by starting to analyze business models of enterprise. Hence, it is clear business models play a vital role in information systems. In this research, we consider the business models with value perspective. Apart from the business models, the term reference model is used in modeling community. We discuss the reference models in the next section.

2.2 Reference Models

Among different kinds of models, the reference model is also highly used in the software engineering context. According to the OASIS (OASIS, 2006), definition a reference model is an abstract framework for understanding significant relationships among entities of some environment. Reference models are also called universal models, generic models, or model patterns (Fettke et al., 2005). In this dissertation, we present generic patterns and models to represent business services. Therefore, we include the concept of reference model in the literature review. The use of reference models has many positive effects for business (Kirchmer, 2009). For example reuse existing reference models as a starting point to develop specific conceptual models. Among different kinds of reference models, the business process reference models are used to develop the enterprise information systems. The process reference model represents dynamic aspects of an enterprise, e.g., activity sequences, organizational activities required to satisfy customer needs, control-flow between activities, particular dependency constraints etc.(Becker et al, 2003). Few examples of process reference models are SAP R/3 (Keller and Teufel. 1998), Enterprise Modeling for Ecommerce (ECOMOD). The survey done by Fettke et al. (2005) shows that some of the reference models are originated from scientific base (e.g., ECOMOD) and some of them are originated from practice (e.g., SAP R/3).

2.3 Business Modeling Ontologies

An ontology is "an explicit specification of a conceptualization" according to Gruber (1993). In other words, ontology explicitly describes a particular domain of interest. Ontologies are increasingly popular tools to identify the basic notions of business models. There exists a number of ontologies (Gordijn et al., 2000; Dietz, 2006; Uschold and Gruninger, 1996; McCarthy, 1982) for business modeling. Among them e^3-value ontology (Gordijn et al., 2000), Resource-Event-Agent (REA) – (McCarthy, 1982) and business model ontology (BMO)-(Osterwalder, 2004) are three established business model ontologies. Initially these ontologies had their own specific purposes but later on they were extended and used in enterprise modeling.

e^3-Value: Gordijn (Gordijn et al., 2000) provides business model framework called e^3-value, which is based on a generic value-oriented ontology. The primary goal of e^3-value is identifying exchange of value objects between the business actors. The basic concepts in e^3-value are actors, resources, value ports, value interfaces, value activities and value transfers. This framework allows the graphical representation and understanding of value flows between the several actors of a model. e^3-value focuses on modeling value networks of cooperating business partners and provides mechanisms for profitability analysis that help in determining whether a certain value network is sustainable

Resource-Event-Agent Ontology: REA was originally formulated as a basis for accounting information systems (McCarthy, 1982) and focused on representing increases and decreases of value in an organization. It was extended further by Geerts and McCarthy (1999); UN/CEFACT (2003); Hruby (2006). A detailed description about REA is available in section 2.3.2.

Business Model Ontology (BMO): BMO has a wider scope with comparison to the above two ontologies. In addition to modeling exchanges of resources, BMO addresses internal capabilities and resource planning. It consists of nine core concepts in four categories Furthermore, BMO incorporates marketing aspects describing value propositions as well as marketing channels (Osterwalder, 2004).

Anderson et al. 2006 presented a common ontology based on above three business ontologies – the e^3-value, REA and BMO. They produced a set of mappings between e^3-value and REA indicating strong similarities between the concepts of the two.

2.3.1 Motivation for Selecting REA

As a basis of modeling business services, we selected REA ontology considering several reasons. According to the Service Dominant (SD) logic approach (Vargo and Lusch, 2004) one significant aspect of service is co-creation of value. Although the e^3-value business model is rich enough to identify value exchanges in a network, value co-creation is not sufficiently addressed. The resources are important ingredients for value co-creation. Therefore, we consider the resource-service relationship can be elaborated more with REA, because resource is a first class citizen of REA.

REA offers a comprehensive ontology of business concepts and it is used in enterprise information systems architectures, frameworks and standards. The authors of Gailly et al. (2008) summarized the applications of REA into several categories. Their analysis is limited to the applications that were proposed or developed after the REA extensions were published in (Geerts and McCarthy, 1999; Geerts and McCarthy, 2002). According to them one ontological application of REA is in model-driven design. The ISO Open-edi specification (ISO/IEC, 2007) uses REA as an ontological framework for specifying the concepts and relationships involved in business transactions and scenarios in the Open-edi sense of those terms. Furthermore the REA ontology definitions are part of the work of UN/CEFACT (United Nations Center for Trade Facilitation and Electronic Business) which is an international e-business standardization body known for its work in the area of electronic data interchange – EDI (UN/CEFACT, 2008).

Considering several features, we select the REA ontology as the basis of the proposed business service modeling framework. The most important two features among them are: value co-creation which is one of main concern of service provisioning, is well expressed with REA. The next feature is, REA provides better insight into resources in a company. Hruby (2008) lists down several advantages of using REA in software development as follows.

- The application design based on the REA model is concise and easy to understand both for the users of software applications and for application developers.
- The same modeling principles are used across all application areas in the business domain
- As REA software applications store the primary data about economic resources, all reports and all accounting artefacts are always consistent, because they are derived from the same data.
- The REA model provides more complete, transparent, and up-to-date reporting for business decision than reporting based on the accounting artefact.

2.3.2 Resource Event Agent (REA) Ontology

The Resource-Event-Agent (REA) ontology was formulated originally in (McCarthy, 1982) and has been developed further, e.g., in Geerts and McCarthy (1999); Gailly et al. (2008); Hruby (2006). The concepts of REA reflect business accounting where the needs of managing businesses through a technique called double-entry book keeping was formerly the standard of use. REA replaces double-entry with semantic models of economic exchanges and conversions. The main focus of REA is representation of increases and decreases of value in an organization. The core concepts in the REA ontology are Resource, Event, and Agent. Figure 2.1 shows the basic concepts of REA.

Agent: An agent is an individual or organization capable of having control over economic resources, and transferring or receiving the control to or from other agents Geerts and McCarthy (1999). REA models are often created from a perspective of a specific company.

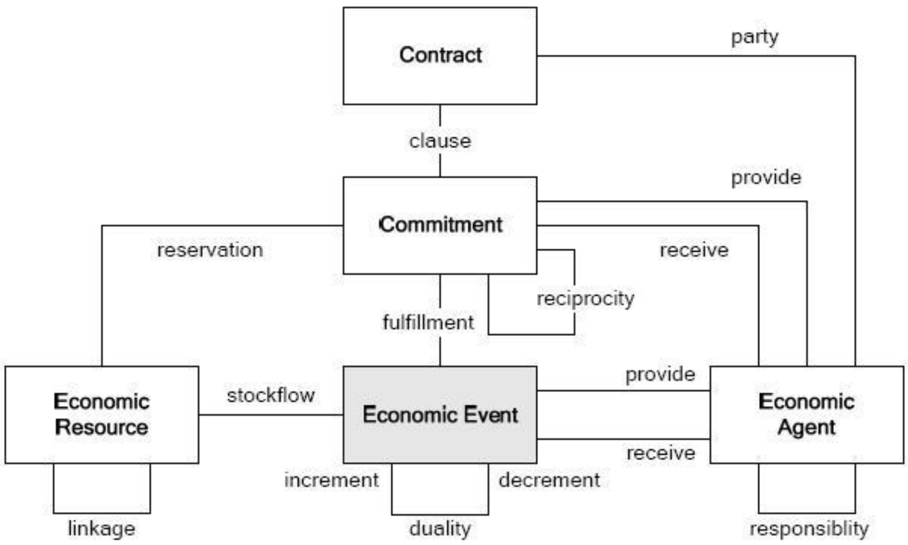

Figure 2-1: Basic REA concepts

Resources: Resources are things that are scarce. Resource is defined as "any object that is of utility and under the control of some enterprise". Originally, only resources that could be exchanged were considered, such as goods, services and money. Later on, internal resources were taken into account as well, including intangible ones like knowledge (ISO/IEC, 2007). Resources are modified or exchanged in processes. A conversion process uses some input resources to produce new or modify existing resources. For example, water and flour can be used as input economic resources in a baking conversion process to produce the output economic resource bread. An exchange process occurs as two agents exchange (external) resources. To acquire a resource an agent has to give up some other resource. For example, in a goods purchase a buying agent has to give up money in order to receive some goods. The amount of money available to the agent is decreased, while the amount of goods is increased. This combination of events is called a *duality* and is an expression of economic reciprocity - an event increasing some resource is always accompanied by an event decreasing another resource.

Event: The constituents of processes are called economic events. An economic event is carried out by an agent and affects a resource. In REA, the notion of stockflow is used to specify in what way an economic event affects a resource. Gailly et al. (2008) extended the basic REA stockflow concept by adding specialization to the stockflow relationship as inflow and outflow. REA identifies five stockflows: produce, use, consume, take and give, where the first three occur in conversion processes and the latter two in exchange processes. The stockflows produce and take are positive stockflows in the sense that they increase the value of some resource for an agent – an economic event with a produce stockflow creates or improves some resource in a conversion process while an economic event with a take stockflow transfers a resource to the agent in an exchange process. Similarly, the stockflows use, consume and give are negative stockflows in the sense that they decrease the value of some resource for an agent – an economic event with a use or consume stockflow uses or consumes some resource in a conversion process while an economic event with a give stockflow transfers a resource from the agent in an exchange process.

Commitment: The commitment is a promise or obligation of economic agents to perform an economic event in the future. For example, line items on a sales order represent commitments to sell goods.

Contract: The contract is a collection of increment and decrement commitments and terms. Under the conditions specified by the terms, a contract can create additional commitments. Thus, the contract can specify what should happen if the commitments are not fulfilled. For example, a sales order is a contract containing commitments to sell goods and to receive payments. The terms of the sales order contract can specify penalties (additional commitments) if the goods or payments have not been received as promised

According to REA a business can be explained with four questions. The authors of Schuster and Motal (2009) summarized that the REA concept basically answer who, what, when and why questions of business collaboration. According to them the REA ontology can be expressed as follows.

- Who is involved in the collaboration (Economic Agents - Buyer, Seller)?

- What is being exchanged in the collaboration (Economic Resources – e.g., Money, Good)?
- When (and under what trading conditions) do the components of the exchange occur (Economic Events – e.g.: Payment, Shipment)?
- Why are the trading partners engaged in the collaboration (duality relationships between resource flows)?

2.4 Business Patterns and Pattern-Based Modeling

The researchers mentioned in section 2.5, follow different approaches to model business activities as services. Some of them identified business services as text based list (Weigand et al., 2009; De Castro et al., 2009). There are some works which included existing business models as they are, but not as services (Arsanjani et al., 2008; SOAML- OMG, 2009). To capture the real business services in an easy way, we propose pattern based service modeling framework. The patterns reduce the design time while assuring the domain concepts. Therefore, we discuss the business patterns and pattern based modeling in this section.

The patterns help to designer to grasp the domain concepts easily. Hence, we investigate the notion of the pattern and the pattern based approaches in business modeling. Alexander presents a general definition to the pattern as follows.

> *"Each pattern describes a problem which occurs over and over again in one environment, and then describes the core of the solution to that problem, in such a way, that you can use this solution a million times over, without ever doing it same way twice "*, Alexander (1977).

Patterns are used in different disciplines as a way to record expert knowledge for problem solving in specific areas Bottoni et al. (2010). For example, design patterns are used in software engineering Gamma et al. (1995), analysis patterns are used in conceptual modeling (Fowler, 1996) and architectural patterns in information system architectures. Business patterns are used as analysis pattern in software engineering.

2.4.1 Business Patterns

Business patterns are reusable models that describe how the companies perform businesses (Hruby, 2006). Designers of business information systems use business patterns to model recurring functionality in a transparent way (Vandenbossche, 2007). There are motivating research works in the literature that uses the business patterns in software design, for example (Hruby, 2006; Marshall, 2000; Arlow and Neustadt, 2003). Following the MDA approach, Hruby (2006) defines more than 20 business patterns which attempt to provide the knowledge about business domain in the form of object-oriented models. He defined REA-based business patterns that describe the structure and the functionality of a business. The structure of the business is described in the form of structural patterns on the operational level and the policy level. Operational level patterns describe the actual economic exchanges and the policy level patterns describe general rules that govern the business events. The business behavioral patterns provide specific functionality that business applications usually have. Hruby's research work presents good guidelines to application developers as well as framework developers.

2.4.2 Pattern-Based Modeling

Patterns are increasingly used in the definition of software frameworks, as well as in Model Driven Development, to indicate parts of required architectures, derive code refactoring, or build model-to-model transformations (Bottoni et al., 2009). The authors of Bottoni et al. (2009) mentioned that the full realization of the power of patterns is hindered by the lack of a standard formalization of the notion of pattern. To overcome the above limitation they proposed a language-independent formal approach to pattern-based modeling which is grounded in category theory (Lane, 1998). This approach supports to define pattern specification, pattern discovery, instantiation, pattern composition, conflict analysis and using patterns in model completion. We follow this formal approach to define the business service patterns and pattern composition in this research.

2.5 Related Work in Service Oriented Design

The evolution of software engineering design has passed through various eras, including structured analysis and design, object oriented analysis and design (OOAD), and component based software design and eventually service-oriented design. To find a solution to our research question, we carried out a literature survey about the service-oriented design approaches which incorporate business thinking.
In Service-Orientation, the key abstraction is the "service". As we mentioned in the chapter one, the term 'service' has different meanings in business and software engineering disciplines. The notion of service as previously described provides an opportunity to increase the flexibility and reuse of business functionality within an enterprise and with partners (Arsanjani et al., 2008). Although the concept of service-oriented architectures (SOA) has been in discussion for several years, it is mainly regarded as a technical concept (Kohlmann and Alt, 2009).

However, to achieve the maximum benefit of SOA, a list of business requirements together with state-of-art technical specification at the design stage is not sufficient. Having realized the importance of service design that allows to communicate with business analysts as well as system designers, several researchers have tried to start the design with business perspectives (De Castro et al., 2009; Weigand et al., 2009; Haesen, 2009). Among several methodological approaches in service modeling, we select a few (SOMA- Arsanjani et al.(2008) , WSMO- Roman et al. (2005), SOD-M - De Castro et al., (2009), SOAML- OMG (2009) and Weigand et al. (2009)), which were based on MDA approach and have incorporated business modeling in their methodologies.

SOMA (Service Oriented Modeling and Architecture):

IBM's SOAD has evolved into SOMA- Service Oriented Modeling and Architecture (Arsanjani et al., 2008) described as a software development life-cycle method invented and initially developed in IBM for designing and building SOA-based solutions. SOMA is a full-blown modeling methodology consisting of three steps: identification, specification, and realization of services, flows (business processes), and components realizing services. The process is highly iterative and incremental. However, because SOMA is proprietary to IBM, its full specification is not available. SOMA also advocates a meet-in-the-middle approach. Domain decomposition is a top-down analysis that starts with analysis of the functional areas in the business domain and of the business processes. This is complemented by a bottom-up asset analysis. The two lines are pulled together by Goal-Service Modeling (GSM). SOMA incorporates many more methods and techniques, including conceptual data modeling, and advocates a fractal model for software development. SOMA links the system level to strategic business goals, but it does not support service modeling at the business level. Because of that, alignment between CIM and PIM level remains limited. Although Goal-Service Modeling is useful, it cannot replace business service modeling and analysis as such.

WSMO (The Web Service Modeling Ontology):

The Web Service Modeling Ontology (WSMO) - Roman et al. (2005) provides a conceptual foundation for describing several aspects related to semantic web services on the web by refining the Web Service Modeling Framework (WSMF), (Fensel and Bussler, 2002). It aims to describe general services which can be accessible through the web service interface with the intention of enabling automation of key tasks (e.g., discovery, selection, composition, etc.). Even though it makes a clear distinction between Services and Web Services, the business service identification and the impact on the resources related to services are not addressed.

SoaML (Service Oriented Architecture Modeling Language):

Service Oriented Architecture Modeling Language – SoaML (OMG, 2009) is an OMG standard which promotes business thinking in its specification. It is an extension of UML to support SOA modeling. SoaML offers a higher level of abstraction of SOA that hides the complexity of lower level Web Services and supports flexible platform choices. It addresses business requirements by integrating the UML business motivation model (BMM). In our view, BMM is one of the best suites to identify business plans and elements of business plans in terms of the business goals and objectives (means and ends in BMM to achieve some business vision, but not the best suite to identify the real business services.

SOD-M (the Service-Oriented Development Method):

SOD-M (De Castro et al., 2009) is a method for the service-oriented development of IS. SOD-M focuses on the development of the behavioral aspect of IS and defines guidelines for building the behavioral models from high-level business modeling. Following a MDA approach, it proposes a set of models starting from CIM level and then extending those to PIM and PSM levels. They consider two different views for the modeling purpose namely business view and information system view. The concepts corresponding to the business view describe the elements inherent to the business, and these elements are represented on the CIM level using the *value model* and the *business process model*. The concepts corresponding to the information system view are elements used to describe the functionalities of the system, and these are represented on the PIM and PSM using *use case model*, *extended use case model*, *service process model*, *service composition model*, *Web service interface model* and *Web service composition model*. Finally, the concepts corresponding to both views, use to align the high-level business models and IS. They use UML notation. In this approach business services are identified on the basis of value models and process models. These identified business services are represented in the form of a textual list. In this way, the resource dependencies between business services cannot be rendered. SOD-M does support the modeling of control dependencies, but in our opinion, this is of limited value as these control dependencies are usually not very stable.

Value-based service design method *(Weigand et al., 2009)**:***

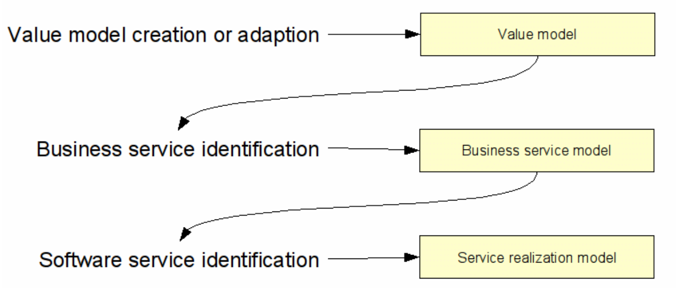

Figure 2-2: Service design method schematic overview

Weigand et al. (Weigand et al., 2009) build on the e^3-value ontology to develop a value-based service design method. The overview of their approach is given in Figure 2-2. This framework suggests to classify business services by categorizing different types of services (core, enhance, and coordination). Using the value model, the different kinds of services are distinguished. Then a table notation is used to describe services which are identified in the previous step (the table consists of four columns: the core service, its enhance services, its coordination services and the applicable policies). Following a top-down approach, the third step helps to identify the services at the informational level and infrastructure level which relates to software services. Even though strong business thinking is incorporated in web service identification, business service modeling with their characteristics is not worked out.

We compare the selected features of the above four approaches. Table 1 shows the results. As the above approaches follow MDA approach, the table shows the corresponding MDA level/s for each approach. Each approach uses other models in their development process. The second attribute of the table is dedicated to represent the other models used in each approach. As the business thinking is a vital input of service design, the next row explains the business service modeling basis of each. To describe the service with more characteristics (for example service-resource relationship and classification of services), we include fourth and fifth rows into the table. The last row shows the modeling notation/s used in above approaches.

Table 2-1: Feature Comparison of selected service modeling approaches

Features	SOMA	SOD-M	WSMO	SOAML	Weigand et al., (2009)
Correspondence with MDA levels	CIM, PIM, PSM	CIM, PIM, PSM	PIM, PSM	CIM, PIM, PSM	CIM, PIM, PSM
Usage of other models		e^3- value model and business process model Service process model using BPMN	Implemented with UML class descriptions	Business process definition model (BPDM)	e^3-value model
Basis for the Business Service Modeling	Goal service modeling	e^3-value model and business process model	User desired goals are captured and listed down in WSML language	Business motivation model	e^3-value model
Consideration of resources – service relationship	No	No	No	No	Partial
Classification of deferent business services.	No	No	No	No	Yes
*Focus is on BS / SS / Both	SS	Both List of BSs is provided	SS	SS	Both List of BSs is provided
Modeling notation	UML	e^3-value and UML	UML, Meta modeling approach,	UML	e^3-value notation

*BS –Business Services, SS - Software Services

We selected several service design approaches which incorporate business thinking into their modeling. The comparison shows that even though the business thinking is included to the selected approaches, business services are not elaborated as such. The De Castro et al. (2009), Weigand et al. (2009) and OMG (2009) approaches use existing business models to identify business services as a text based list. On the other hand the goals of business models such as e^3-value network, Business Motivation Model are not identifying nor design business services.

Several researches conducted comparisons of service analysis and design approaches using. Among them, Kohlborn et al. (2009) compare 30 service analysis approaches. They also consider usage of SOA concepts (business services, software service or both) as one feature. The other features compared in their study are life cycle coverage, delivery strategy (top-down, bottom-up or meet-in-the-middle), degree of prescriptive and the accessibility and validity. They reveal that a comprehensive approach to the identification and analysis of both business and supporting software services is missing.

Chapter 3

Service Modeling Language

Modeling plays an important role in the software development process. In software engineering, models are used to describe both the problem (requirements) and the solution (design) in order to gain a better understanding of the issues involved. Models must be presented using a modeling language or notation. A language consists of syntax and semantics. The models are also needed to provide abstractions that are adequate for modeling a large system, while ensuring sufficient detail for establishing properties of interest. We introduce a new modeling notation to the proposed service modeling language. However, instead of introducing a new modeling notation, it is possible to use other notations, in particular UML diagrams (UML, 2010) with stereotypes. As UML is a universal language, UML diagrams with stereotypes approach have added advantage. The advantage of BSRM is that the different concepts are easier to recognize because of the different shapes.

One major pillar of the service modeling framework is the service modeling language. The State of the Art of the service modeling languages with business perspective, which is described in chapter 2, reveals the deficiencies and the strengths of the existing approaches. The need of service modeling language with business perspective is very clear. Therefore, strong business modeling basis is an essential requirement for the proposed service modeling language.

Selecting proper business ontology to define the service modeling language is one of the major challenges of this research. As we stated in chapter 2, value creation has always been the most important business concern. Recently, business researchers have started to recognize the intimate relationship between value creation and service provisioning. As described in the chapter 1, S-D logic proposed by Vargo and colleagues (Vargo and Lusch, 2004; Alter, 2008) focuses on service provision in contrast to goods production (G-D logic). The shift from G-D to S-D logic is one from a value proposition consisting of operand (passive) resources to one consisting of operant (active) resources. Instead of only seeing value being created within companies that exchange the means for this value creation from one to another, it emphasizes the value being created between companies (or companies and consumers). Grönroos (2008) goes even further by saying that value is primarily created at and by the customer, and the company is just co-creator. Although the e^3-value business model is rich enough to identify value exchanges in a network, value co-creation is not addressed well. Hence for this research we make use of one comprehensive and well established business model ontology, the REA ontology (McCarthy, 1982; Geerts and McCarthy, 1999; Hruby, (2006) but also use some concepts from Dietz' enterprise ontology Dietz (2006). As we described in chapter 2, the original REA ontology mainly treated for accounting information model. Later on it has been extended further, (e.g., Geerts and McCarthy, 1999; Gailly et al., 2008; Hruby, 2006) into a comprehensive enterprise architecture.

The main objective of this chapter is to introduce the service modeling language called Business Service and Resource Model (BSRM). The constructs of the language are based on the metamodel which is grounded in REA and its extension. Therefore, we describe the Metamodel for the service modeling language first. The metamodel section has a subsection called BSRM Language Semantics, to describe the semantics and the axioms of the modeling language. The implementation of the metamodel using ConceptBase tool is described in section 3.2. Then we move to the modeling notation and the Service Design Examples are described accordingly. The BSRM Viability Check subsection contains, among others, a comparison of selected features of BSRM with other models. Finally, this chapter ends with the subsection called Service Model Analysis which gives a general overview of value analysis of BSRM.

3.1 Metamodel

The proposed service model was designed to contain a minimal set of concepts and their relations which can be easily grasped by the users. Metamodeling is capable to define the constructs of modeling notations as well as their interrelation to constructs of other modeling notations. We illustrate the metamodel as UML class diagram while internally we represent them in ConceptBase (Jeusfeld et al., 2009). The metamodel which is grounded on Resource-Event-Agent (REA) ontology, for the proposed BSRM-service model is depicted in the Figure 3-1. In the REA ontology, type images are used to represent the intangible structure of economic phenomena (Geerts and McCarthy, 2004). The metamodel described in this chapter is also followed the typification which captures description of group of actual phenomena. The word 'type' is sometimes omitted to reduce repetition in the text.

All the constructs of the proposed BSRM language are defined in the metamodel. Economic resource type is the central spot of the metamodel. The specialisations of the economic resource type and the different types of relationships are defined. The constraints of the metamodel and all the definitions of the concepts are defined below.

The constraints of the metamodel:

- Economic Resource type can be specialised as a Service type, a Physical resource type or an Intentional resource type. The generalization set is expressed as {incomplete, disjoint}, which means that the Economic Resource type can either be a Service Type, a Physical resource type, an intentional resource type or none of the above but not falls into more than one category. According to REA, the resource is a thing that is scarce, and has utility for economic agents, and is something users of business applications want to plan, monitor, and control. (ISP/IEC, 2007) defines the economic resource type as an abstract specification of an economic resource where its grouped properties can be designated without attachment to an actual, specific economic resource which is classified as a good, a right or a service of value, under the control of a person. Following the same perspective of economic resource, we distinguish three separate type of economic resource type as physical resource type, service type and intentional resource type.
- Each Service type is either an Exchange service type or a Conversion service type. The generalization set is expressed as {complete, disjoint}, which means

that the Service type must be either Exchange or Conversion and not both. The service type has three roles with itself namely enhance, coordinate and part-of. The Enhance service is a service that aims to increase the value of another service. In other words, the enhance service converts another service into something more valuable. Hence all the enhance services are conversion services. If the coordination service involves both the focal and an external agent (typically when the coordination service supports an exchange service), then we classify this coordination service as an exchange service. If the coordination service involves services other than exchange, then we classify this coordination service as a conversion service. The services which have part-of relationship with core-service are called sub-services. These services fall in to either conversion or exchange as those sub services are part of the core service.

- Each Stockflow type is either an Inflow type or an Outflow type. The generalization set is expressed as {complete, disjoint}, which means that the Stockflow type must be either Inflow or Outflow and not both. According to REA, all the stockflow types are categorised into two as inflow and outflow. Inflow type has two specializations and outflow type has three specializations.

- Each Agent type is either a Focal agent type or an External agent type. The generalization set is expressed as {complete, disjoint}, which means that the Agent type must be either Focal or External and not both.

Metamodel

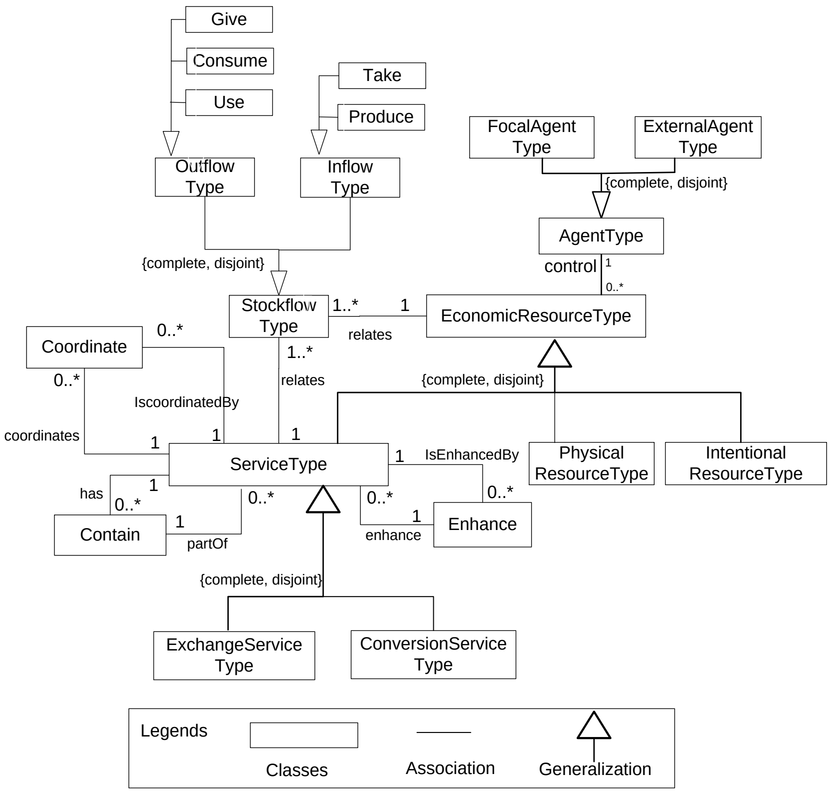

Figure 3-1: Metamodel for BSRM (UML style)

Definitions of the concepts in the metamodel:

Economic Resource Type: Economic resource type is an abstract specification of an economic resource where its grouped properties can be designated without attachment to an actual, specific economic resource (ISO/IEC, 2007). Economic resource is one of the key concepts in REA. REA defines the economic resource as a good, right or service of value, under the control of a Person. The resource-based view of the firm (RBV) which is build on the works of Wernerfelt (1984) and Barney (1991), is one of the popular strategic management concepts to be enthusiastically embraced by marketing scholars (Fahy and Smithee, 1999). The resource-based view defines the resource quite broadly. The resources include human, financial, tangible (such as plant, equipment, buildings) and intangible resources (such as patent. know-how, brand names, experience and organizational routines) (Douma and Schreuder, 2008; Caves, 1980). The notation of the resource in REA and RBV has some commonalities. But resource-based theory gives a broader view to the resource which is more aligning with strategic choices. According RBV, a resource can be the basis of competitive advantage only if that resource has following characteristics: valuable (if it is difficult to buy due to its specialty), barriers to duplication, rare and non-substitutable (Douma and Schreuder, 2008). Therefore, the RBV emphasizes strategic choice of identifying, developing and deploying key resources to maximize returns. But in this thesis we consider the economic resource as defined in REA and distinguish three subtypes of economic resources namely service types *service type, physical resource type* and *intentional resource type.* Each of them is described below.

Service Type: As a service is an object that is considered valuable by actors and that can be transferred from one actor to another. For example the transport service provided by the transportation company to the passengers. In (ISO/IEC, 2007), the service is defined as a specialization of economic resource. However, in a service-oriented business perspective, a service is more; a service is a value creating process with operand and operant resources. In REA, value is created in conversion processes and in exchange processes. Hence we distinguish two service specializations: exchange service and conversion service.

Conversion Service: The simple definition of "conversion" is transform input into output. The *conversion* service corresponds to a group of decrement and increment economic events in REA.

Exchange Services: Exchange is giving something valuable to other party in return of another valuable thing. The *exchange* service corresponds to a group of decrement and increment economic events in REA. As we discussed in the constraints of coordination service in metamodel, the coordination service can be an exchange service when the focal agent and external agent participate to the exchange. Then the exchangeable thing can be an intentional resource.

Physical Resource: Physical resource is another economic resource is a type that appears in what Dietz (2006) calls the production world. According to ISO/IEC (2007), physical resources are called *Goods* which are tangible and several examples of physical resources are also listed in the above reference. Such as materials, capital assets, real states or funds like money. The human resource is a special type of resource. We distinguished two types of role in human resource i.e. active participation and passive participation. The active participation of the human resource in value creation process is provisioning of human's service. For instance, the heir cutter's service in the heir cutting service. The detail discussion of the human resource provisioning is available in chapter 5. We consider the passive participation of human resource as a physical resource. For instance, the customer in the heir cutting service.

Intentional Resources: Intentional resources that appear in the social or coordination world, which are not material and typically need representation by means of informational objects or an intentional resource, can be a behavioral or psychological attribute or skill of a human being. For example, motivation towards to a work. Usually, the intentional resource has reference to a physical resource. For example, *Sales Order*, which refers to the products in the sale. In REA, the intentional resources are policy-level objects.

For further characterize a service, we need to define the possible relationships among the services and with other resources. The service-oriented literature typically distinguishes one basic relationship, "use" - one web service (customer) uses another (provider). When a composite service "uses" a couple of atomic services, the latter can also be called sub-services and the relationship can be seen as aggregation. In our view, the "use"- relationship lacks sufficient semantic precision. More specific relations need to be distinguished.

Stockflow: REA defines the stockflow as the relationship between Resource Type and the Event Type. It is lifted in BSRM to a relationship between Economic Resource Type and Service Type. Gailly et al. (2008) extended the basic REA stockflow concept by adding specialization to the stockflow relationship as inflow and outflow. We adopt this specialization into our model. An inflow relationship from resource to service means that the resource is used or consumed (value decrement), whereas an outflow from service to resource means that the resource is produced or its value is incremented. At instance level inflow can be produce and take relationships and outflow can be use, consume or give. At instance level for any exchange service there must be at least one inflow and one outflow of resources, which represents the exchange duality; and similar for the conversion services.

Core Service: The service classification model developed in Weigand et al. (2009), categorizes services into several categories. According to this paper, starting point of service classification is the recognition core services. The reason for viewing these services as core is that they provide the raison d'être for an actor in a value network, as they specify what value the actor is able to provide to the network. Core services are easy to identify. We adopt the same idea to our service modeling. The core- service is not visible in the metamodel in its name. But it refers to the central service that appears as exchange or conversion service in physical world. Given a set of core services, there is a need for a number of services that add to or can improve on the core services. Among these service categories enhance services and the sub-services which have part of relationship to the core services are special types of services that play a major role in service modeling at business level. These service categories correspond to different roles of the service type in our model.

Enhance Service: Enhance service is any service that adds value to any other service called its goal. An example of an enhance service is a service that advertises another service, or manages it. In terms of REA, enhance corresponds to a stock-outflow ("produce") relationship between two services, for instance, a management service and an operational service.

Sub-Service: Sub-service (part-of) is motivated as follows. Assume that we have a service A, whose realization involves multiple value activities and it makes sense (economically) to view these value activities as independent services that are shared by different contexts. These services are called sub-services of A. There are two categories of sub-services namely core-sub services and coordination services.

> *Core-sub services*: The first category of the sub-service called core sub-service, is defined as a service that is used somewhere in the realization of A by manipulating physical resources. In terms of REA, there exists a stock-inflow relationship between the core sub-service and the composite service. An example of a core sub-service is hair washing as part of hair-dressing.
>
> *Coordination services*: When there are multiple sub-services, their dependencies need to be coordinated (Malone et al., 1994; Schmidt and Simone, 1996). The other category of sub-service called coordination service is defined as an enhance sub-service: if B coordinates A, then B enhances A (A is the goal of B) and it is

used somewhere in the realization of A. Coordination services do their job by manipulating intentional resources.

Agent: Following REA, we include the *agent* type concept to the metamodel. An agent is an individual or organization capable of having control over economic resources. Agent type is classified into two, namely focal and external. The first category considers the perspective of one of the actors of the value network, to be called the focal agent, and identifies services needed by that actor (Weigand et al., 2009). The focal agent type is the individual or the central organization who intends to view its business with service perspective. The focal agent type is more similar to one trading partner who participates business activities and views intra-organizational and inter-organizational business activities as his own view (ISO/IEC, 2007). On the other hand the independent viewer's perspective, who views the business activities as an independent viewer, is considered as External agents. The BSRM model always takes the perspective of a Focal agent. The design decision to deemphasize the agents/owners is in line with the SD-logic approach in which co-creation is more important than ownership.

As pointed out above, a service can be exchanged between two agents in the same way as goods, by means of an exchange service. The distinction between the service exchanged (e.g., a hair-dressing, or a flight) and the service that exchanges it is an important "separation of concerns" principle. For coordination services supporting an exchange we suggest the use of generic services wherever possible. These can be related to business transaction phases such as distinguished in UN/CEFACT, (2003). In Weigand et al. (2010), a REA based characterization of coordination service is given in which such a service is defined on the basis of the coordination object it produces, such as contract (often called Purchase Order in practice) or reservation,

Some remarks on the design choices for BSRM:

- The BSRM ontology does not contain events. This may come as a surprise, especially as BSRM is based on REA. However, this design choice is a consequence of our service-oriented perspective. Resource dynamics are captured under conversion and exchange services. From the duality principle of REA it follows that all events can be captured this way, so they are not especially needed in BRSM.

- Most service modeling approaches focus on process aspects and service value. We have included resources. Not only because they are needed to describe service effects and for the data modeling part, but also because resources play a prominent role in new service design (Froehle and Roth, 2007).

- In contrast to most other approaches that only consider one type of relationship between services, we identify as many service linkages (five relationships as stock flow type and three as different roles of service type) as possible, and keeping in mind the tenet that meaning is captured in structural relationships.

- BSRM does not aim to be an all-encompassing language, but only identifies the service model core. However, the core should be rich enough to allow mapping with complementary models such as process model and value model.

- The physical and intentional resource types in BSRM appear only once in the model for a particular business activity. But it may have several specialisations when completing the business activity. For example, the delivery service which delivers the customers goods. In that case the *delivery* is the service which uses the *goods* and increases value by changing the location of the goods. In the BSRM model, the *goods* appear only once and it has "*use*" and "*produce*" relationship with the *delivery* service. It is possible to depict the good as two different states, "*good*" and "*delivered good*".

3.1.1 BSRM Language Axioms

The BSRM metamodel provides comprehensive definitions to the new language, but to complete the semantic definition, we need to specify additional axioms. For the ease of understanding, we write them down in plain English.

- *service stockflow completeness.* For each service, at least one inflow and outflow resource should be specified. This axiom derives from the REA duality principle.

- *physical resource flow completeness.* For each internal physical resource type, at least one inflow and outflow service should be specified. For external ones, this is not required (as the model is bounded in scope). Although we should allow for partial models where this axiom is not obeyed, for practical reasons, this completeness axiom is important for the underlying ontology. Resources that are used should also be replenished somehow. Resources that are not used in some service do not add value, apparently, and so they should be skipped.

- *enhancement completeness.* Enhance services, including coordination services, may have several inflow and outflow resources, but they should always have at least one inflow and one outflow of the intentional type.

- *intentional resource completeness*. It is a characteristic of intentional resources that they are not consumed by use. Still, we require that for each internal intentional resource an inflow service is specified, otherwise the intentional resource cannot co-evolve with the system. If it does not co-evolve, it is considered to be external. Intentional resources should also have an outflow service specified in order to be relevant, and this service should be an enhance or coordination service. The latter constraint makes it possible to relate the intentional resource indirectly to a physical resource or physical resource manipulating service. Without such a relationship, the intentionality characteristic would be questionable.

3.2 Implementing BSRM with ConceptBase

The objective of implementing the BSRM metamodel and models with a modeling tool is twofold. The first objective is providing a fast mechanism to build the model. The second objective is automatic validation of the models based on the rules and definitions defined in the metamodel. We select ConceptBase as the modeling tool.

ConceptBase (Jeusfeld et al., 2009) is a tool for conceptual modeling and metamodeling. It supports unlimited meta class hierarchies by representing information at the data level, class level (example, schemas), meta class level (modeling language constructs) and meta-meta class level (constructs used to define modeling languages). The graphical view of the models is another attractive feature of ConceptBase. All these reasons lead us to select ConceptBase as the implementation tool of BSRM metamodel and the modeling language. For example by following OMG MOF abstraction levels M0 level represents the data, M1 level represents model or schema, M2 level represents the notation, M3 level represents the notation definition, and so forth. In our approach the BSRM metamodel represents the M3 level and modeling language represents theM2 level.

For BSRM- service metamodel, we have the following definitions. Concept is the basic element in ConceptBase. The concepts and their relationships of the service metamodel are defined in the following code. Economic resource type is the central concept of the service metamodel. First block of code defines the economic resource type and its possible relationships. Second block of code defines the Service type is which is a specialization of an economic resource type with its relationships to coordinate, enhance, part of services. The rest of the code defines the all the concepts and the relationships with constraints which are defined in the service metamodel.

```
EconomicResourceType in Concept with
  connectedTo
     stockflow: EconomicResourceType;
     flowToService: ServiceType;
     flowToSingleService: ServiceType;
     flowToMultiService: ServiceType
  end

AgentType in Concept with
  connectedTo
    control: EconomicResourceType
end

ServiceType in Concept isA EconomicResourceType with
  connectedTo
     coordinate: ServiceType;
     flowToResource: EconomicResourceType;
     enhance: ServiceType;
     partOf: ServiceType
     take: EconomicResourceType;
     produce:EconomicResourceType;
     give: EconomicResourceType;
     use: EconomicResourceType;
     consume:EconomicResourceType
end
```

```
PhysicalResourceType in Concept isA EconomicResourceType end
IntentionalResourceType in Concept isA EconomicResourceType end
```

The constraints for the stockflows are defined as active rules.

```
ECArule Take_Map with
  ecarule
        er : $  s/ServiceType r/EconomicResourceType
        ON Tell (s take r)
        IF TRUE
        DO Tell (r flowToService s)
        $
end

ECArule Give_Map with
  ecarule
        er : $   s/ServiceType r/EconomicResourceType
        ON Tell (s give r)
        IF TRUE
        DO Tell (s flowToResource r) $
end

ECArule Produce_Map with
  ecarule
        er : $   s/ServiceType r/EconomicResourceType
        ON Tell (s produce r)
        IF TRUE
        DO Tell (s flowToResource r) $
end

ECArule Consume_Map with
  ecarule
        er : $   s/ServiceType r/EconomicResourceType
        ON Tell (s consume r)
        IF TRUE
        DO Tell (r flowToService s) $
end

ECArule Use_Map with
  ecarule
        er : $   s/ServiceType r/EconomicResourceType
        ON Tell (s use r)
        IF TRUE
        DO Tell (r flowToService s) $
end
```

The constraints are also defined as query classes.

```
FlowServiceType in QueryClass isA ServiceType with
   constraint
       hasInOut: $ exists r1,r2/EconomicResourceType
       (this stockflow r1) and (r2 stockflow this) $
End

{* a coordination service is simply a service that coordinates another
service *}
CoordinationServiceType in QueryClass isA ServiceType with
```

```
   constraint
     csupp: $ exists s/ServiceType (this coordinate s) $
end

{* an enhance service is a service that enhances another service *}
EnhanceServiceType in QueryClass isA ServiceType with
   constraint
     csupp: $ exists s/ServiceType (this enhance s) $
end
```

The complete code is available in appendix A.

3.3 BSRM Modeling Notation

In this section, we propose simple modeling notations to the metamodel introduced in the section 3.1. For the clear differentiation between services and the other resources we use different symbols. Services are denoted as rounded rectangles (symbol of terminator) and other resources are denoted as square rectangles. We differentiate services further; for *Conversion* service we filled with a color and for the Exchange service the label in the box is ended with the word *Exchange*. The relationships show the flow of the resources by an arrow head. From the resource perspective, the inflow represents the "produce" or "take" relationship and the outflow represents "use", "consume" or "give" relationship. We distinguish "use" and "consume" relationships separately. "Use" of resource means the resource is allocated for particular period for one activity and then it is released. On the other hand if a resource is consumed, the allocated resource not exists at the end of the service completion. Hence we represent these two relationships with two different notations. The "use" relationship is represented as a dashed line ended with arrow symbol. All kind of "part-of" relation is rendered as a line ended with diamond symbol (following UML) and the enhance relationship by a dashed arrow with a + mark. As far as terminology is concerned, we avoid adding the word "service" to each service name. A summary of the modeling notation is given in table 2.

Note that agents are not explicitly represented in the BSRM model. A BSRM model always takes the perspective of a particular agent, so that the "give" and "take" stockflows are unambiguous. However, this does not mean that all resource types in a BSRM model are owned by the focal agent, as it is considered important to model the effect of a service on resources of the customer. The design decision to deemphasize the agents/owners is in line with the SD-logic approach in which co-creation is more important than ownership.

As mentioned in the introduction of this chapter, instead of introducing a new modeling notation, it is possible to use other notations, in particular UML diagrams with stereotypes. UML approach has added advantage as UML is a universal language. In this research, we identify several service categories, their roles and the relationships with services/resources. The advantage of BSRM is that the different categories and concepts are easier to recognize because of the different shapes.

Table 3-1: Summary of the modeling notation

Service is denoted as a rectangle with rounded widths (Conversion services are filled with a color and exchange services are not filled with a color.)	
Physical resource is denoted as rectangles	
Intentional resource is denoted as dashed rectangles	
Part of Relationship Co-sub Service relationship are denoted as Coordination relationship is denoted as	 +
Enhancing relationship is denoted as	+
Stockflow relationship "Use" Stock outflow relationships is denoted as Stock outflow relationships (consume/give) are denoted as Stock outflow relationships (produce/take) are denoted as	

3.4 Service Design Examples

This section illustrates two simplified examples using BSRM. Figure 3-2(a) depicts the case of sales of goods. This can be modeled as the *ProductExchange* service. The *ProductExchange* service has *give* and *take* relationships with *Product* and *Money* accordingly. As the value of the *Product* is decreased after the exchange, the arrow head is out from the *Product*. For the *Money* it is opposite (arrow head is inwards to the *Money*). That is because of the value of the money is increased after the exchanged. The exchange service is initiated by *SalesOrderProcessing* which is a sub-service more specifically a coordination sub-service. It uses *SalesOrder* which is an intentional resource. *YellowPage* is acting as enhance service to the exchange service by increasing its publicity.

Figure 3-2 (b) shows the simplest version of production. The *Produce* service is conversion service type and has a *use* relation with *Part* and *Tool*. The *Produce* service has "*produce*" relation to the P*roduct*. We define human resource involvement in the production as an enhance service-*EmployeeServiceProvisioning*. (more detail discussion about human resource involvement is available in chapter 5.) *ProductionOrderProcessing* is a coordination sub-service to the *Produce* service. The *ProductionOrder* acts as an intentional resource which is used by *ProductionOrderProcessing.*

The two examples represent in fact quite general patterns (cf. Hruby, 2006). BSRM encourages service modelers to build their models by combining generic patterns and specializing them to the domain in question. The development of such a pattern library is described in chapter 4.

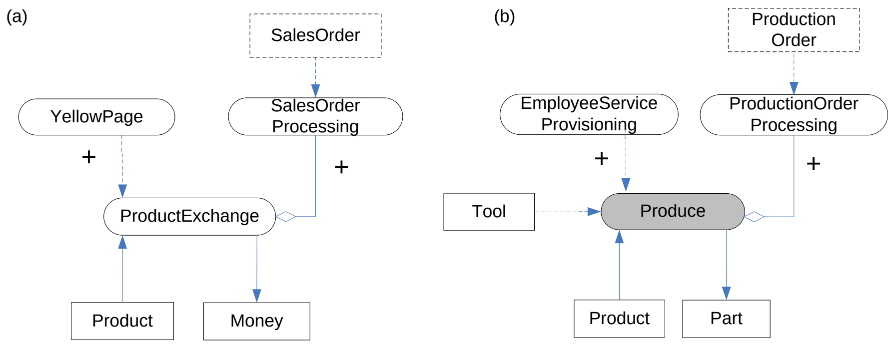

Figure 3-2: Product exchange service and Produce conversion service

We demonstrate the implementation of the model level using ConceptBase we illustrate a part of the sample code in ConceptBase for the ProductExchange service. This is a representation of meta-meta class level (M1 level).

```
YellowPage in ServiceType with
  enhance
    service1: ProductExchange
end

SalesOrderProcessing in ServiceType with
  coordinate
     service1: ProductExchange
end

Product in IndividualizedResourceType with
  flowToService
     flow1: ProductExchange
end
ProductExchange in ServiceType with
  flowToResource
     flow2: Money
end

SalesOrder in IntentionalResourceType with
  flowToService
     flow3: SalesOrderProcessing
end
```

Based on the main concepts which were defined in the M2 level (see sub section 3.2) we derive the M1 level constructs. The first block of code defines the *YellowPage* service. *YellowPage* is a service type and it has *enhance* relationship with *ProductExchange* service. The rest of the codes define the services and resources with their relationship to given example.

3.5 Model Synchronization

A model is a set of rules or procedures for representing a phenomenon. Different models represent different aspects. The BSRM provides service-oriented view at business level. In line with MDA approach, the BSRM represents the CIM level which describes the environment of the system is used. In system development, it is generally accepted that it is not feasible to put all relevant concerns in a single model. As BSRM is new modeling language, it is useful to investigate the semantic relevancies with other models. Hence, we carry out model synchronization in two directions. The first one is the horizontal synchronization which is performed in same level (CIM) of models in particular ERD and e^3-value model. And the next synchronization is vertical in which the mapping between CIM and PIM levels, in particular the BSRM is mapped with BPMN. The intention of synchronization of BSRM model with ERD and e^3-value network models is not converting BSRM into another model, but assessing its possible relevance with other models. The BSRM and business process mapping generates the abstract processes.

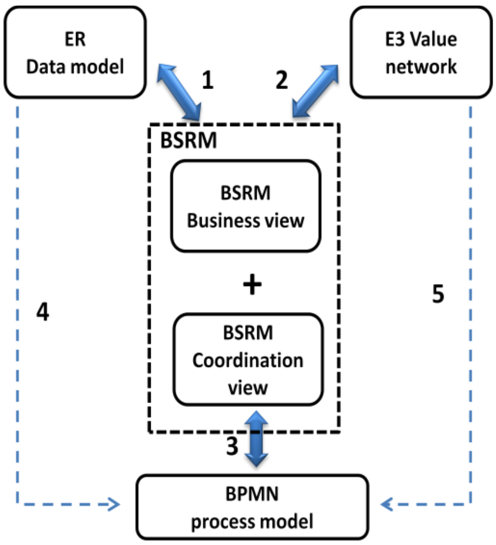

Figure 3-3: Service model mapping

Among the several concerns in system development, data modeling is a basic issue to be addressed. Hence we selected ERD to represent the data aspect. As the e^3-value network is widely accepted business modeling ontology, we considered BSRM and e^3-value network synchronization as well. However, when multiple models are used, they need to be used in a consistent way. In this synchronization process, the BSRM model can be used as central hub (Figure 3-3). The horizontal synchronization in which, the model synchronization in between BSRM model with ERD (arrow 1 in the diagram) and e^3-value model (arrow 2 in the diagram) is direct. These two synchronizations are discussed in section 3.5.1 and 3.5.2 accordingly. The vertical mapping where the BSRM is mapped with process model is discussed in 3.5.3. As this mapping is not direct, we consider the coordination view of the BSRM. The link between ER model and the process model (dotted arrow 3) is not a mapping but finding out possible correspondences between two models. The link between e^3-value model and the process model (dotted arrow 5) also has the same objective as previous. These two links which are represented by arrow 4 and 5 provide additional information when generating the process model from the BSRM coordination view.

3.5.1 Data Model Perspective

The business service model mapping with conceptual data models is important for several reasons. Identifying data stores is one of outcome of data model analysis. These data stores also reoccur in the BPMN process models. Data model analysis may also feedback into service model. For example, many-to-many relationships in data model reveals the need of new entity which are not visible in the service model. We identified general rules for the model mapping that are in line with the current practice of REA/ER mapping (Romney and Steinbart, 2008).

R1: For every Service Type in the service model there is an Entity Type in ER model.
R2: For every Physical Resource Type in the service model there is an Entity Type in ER model. For the Resource Type “Money”, there is both an Entity Type MoneyContainer and a related EntityType MoneyTransfer.
R3: For every Intentional Resource there is an EntityType in ER model.
R4: Stockflow relationships correspond to a (weak) EntityType (“Relationship” Type in classical ER), labeled give, take, use, or produce. This EntityType may get an attribute Period and an attribute Amount.
R5: Partof, and enhance relationships correspond to an Entity Type (“Relationship” Type in classical ER). The labeling must be made on the basis of domain knowledge, e.g., “reserve”, “manage”, “publish”.

About R2: it is clear that every physical resource type that can be individualized (e.g., driver, truck) corresponds to an entity type. For physical resources that cannot be individualized (or dealt with as non-specific) it must be assumed there are domain-specific Container entities with a “level” attribute. We take this to be part of the ER modeling task. Since Money is a resource type that occurs in every exchange, we have formulated a special rule for this case only. In the ER model there must be a money Container type, like Bank Account, and there is a Payment event that increases or decreases the Container. This Payment may refer to multiple exchanges, e.g., if the Customer pays all his bills once per month.

About R4: Conversion inflows are either use or consume. In the case of “use”, the resource is allocated to the service for some Period. If the related ResourceType is not specific, then a certain Amount of the ResourceType is affected.

Example:
Figure 3-4(A) models the “*transport exchange*” service for a transportation company. Transport exchange service gives *transport* service and takes *money*. The transport service uses *trucks*. Upper part the Figure 3-4(B) shows the initial ER model derived, to which connectivity is added (classical ER notation). Conceptual data modeling includes classification of the relationship types in terms of connectivity. This information cannot be derived from the service model.

The lower part of Figure 3-4(B) illustrates how the data modeling may potentially feedback into the service model. The relationships which have many-to-many connectivity are decomposition points in the ER model. In the cargo transport domain, a truck can carry multiple packages at the same time, and a transport between source and destination may have to be split up in many intermediate drives. Note that in other transport domains, e.g., the taxi domain; the situation may be quite different. For the truck transport domain, at least one extra EntityType Truck Drive must be introduced. It is useful to feed this back to the service model, because it signals that a "drive" may be a distinct service (that may even be offered by external "carrier" agents).

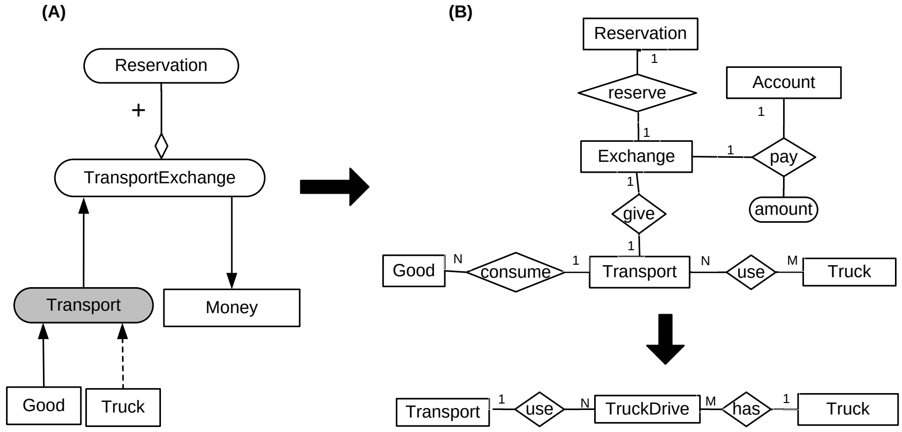

Figure 3-4: Transport- exchange service mode and ER model

To represent the mapping rules in ConceptBase, we first need to represent the source and target notations (BSRM, ER) as meta-classes. Whenever possible, we use so-called query classes to define the constructs. A query class is the deductive counter-part of a defined concept in description logic (Staudt et al., (1993); Borgida, (1996); Baader et al., 2003). Its extension is all objects that fulfill its membership condition (sufficient condition). All other constructs are represented as ordinary meta-classes, i.e. their membership conditions are necessary. For BSRM, we have the following definitions (excerpt):

```
ServiceType in Concept isA EconomicResourceType with
     connectedTo
      coordinate: ServiceType;
      flowToResource: EconomicResourceType;
      enhance: ServiceType;
      partOf: ServiceType
end
FlowServiceType in QueryClass isA ServiceType with
     constraint
      hasInOut: $ exists r1,r2/EconomicResourceType
      (this stockflow r1) and (r2 stockflow this) $
end
  SellServiceType in QueryClass isA FlowServiceType with
```

```
   constraint
       isSell:  $ (forall rout/EconomicResourceType
       (this stockflow rout) ==> (rout in MoneyType) ) and
       (forall rin/EconomicResourceType (rin stockflow this) ==>
       (rin in GoodsType) ) $
end
```

The mapping rules are also defined as query classes. Formally, they return the elements of the BSRM model that are correctly mapped to the target model (here the ER diagram). We include the representation of the rules R1 and R2:

```
EconomicResourceType with
  connectedTo
     storedAs: EntityType
end
EconomicResourceType!stockflow
  connectedTo
     storedAs: EntityType
end
R1_ServiceType_WithEntityType in QueryClass isA ServiceType with
  constraint
     r1: $ exists et/EntityType (this storedAs et) $
end
R21_MoneyType_WithNumberAttribute in QueryClass isA MoneyType with
  constraint
    r4: $ exists et/EntityType d/NumberDomain (this storedAs et)
            and (et entAttr/amount d) $
end
```

The first two frames define the 'storedAs' link between the BSRM and the ER notations. This link is used to represent a mapping between the two notations. The third frame is the query class for rule R1. BSRM, each ServiceType shall be linked to an entity type being its counterpart on the data model. Rule R2 is mapped to several query classes, of which we only display the first one R21: a money resource is mapped to an entity type that has an amount attribute.

The logical-based representation allows not only to retrieve those BSRM model elements that have been correctly mapped, but also to retrieve the unmapped elements, e.g., by:

```
(x in MoneyType) and not (x in R21_MoneyType_WithNumberAttribute)
```

This condition can be used in active rules that actually generate the required ER elements. The active rule specifies certain actions that have to be executed if an event occurs.

3.5.2 The Value Network Perspective

In this section we map the service model with the e^3-value business model. The e^3-value value ontology Gordijn et al. (2000) aims at identifying exchanges of resources between actors in a business case and it also supports profitability analyses of business cases. The basic concepts in e^3-value are actors, resources, value ports, value interfaces, value activities and value transfers (Figure 3-5).

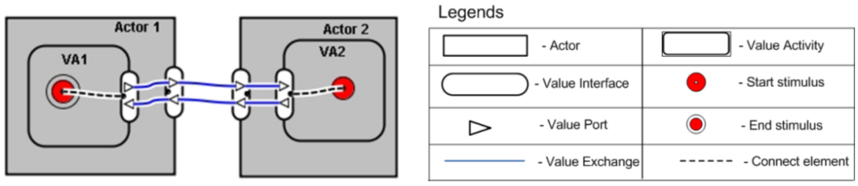

Figure 3-5: Basic e^3-value concepts

An actor is an economically independent entity. A set of actors can be grouped into a market segment. A resource (also called value object) is something that is of economic value for at least one actor, e.g., a service. A value port is used by an actor to provide or receive resources to or from other actors. A value port has a direction: in (e.g., receive goods) or out (e.g., make a payment), indicating whether a resource flows into or out from the actor. A value interface consists of in and out ports that belong to the same actor. Value interfaces are used to model economic reciprocity and bundling. A value exchange represents one or more potential trades of resources between these value ports. A value activity is an operation that can be carried out in an economically profitable way for at least one actor.

Example:

We use the same example used in previous section ("*transport exchange*" service for a transportation company). Figure 3-6 illustrates value modeling and a mapping from the service model. The (implicit) focal agent of the BSRM model is mapped to an e^3-value actor. The value interface corresponds directly to an exchange service (which also induces another e^3-value actor to be added). For every in-port in a value interface in the e^3-value model, there is a take relation in the exchange. Each out-port in a value interface corresponds to a give relation in the exchange service. Value objects, which are depicted as a label on the arrow in e^3-value, correspond to the economic resource types that are given or taken in the exchange. In principle, value activities can be mapped with conversion services. However, the relationships that BSRM represents between conversion services and resources cannot be represented in e^3-value, so we have omitted this mapping.

We derived the following mapping rules for the BSRM – e^3-value network mapping.

R1: For every exchange type service in BSRM, there is a value interface in the value model.
R2: For every "take" and "give" relationships in service model, there are "in" and "out" ports respectively, in the value model.
R3: For every physical resource type in service model, there is a value object in the value model.
R4: For conversion type service in service model, there is a value activity in the value model.
R5: Intentional resources and coordination services are left out of the value model

model, unless they are offered by a third party. In that case, there is a value exchange in which the coordination service is a value object.

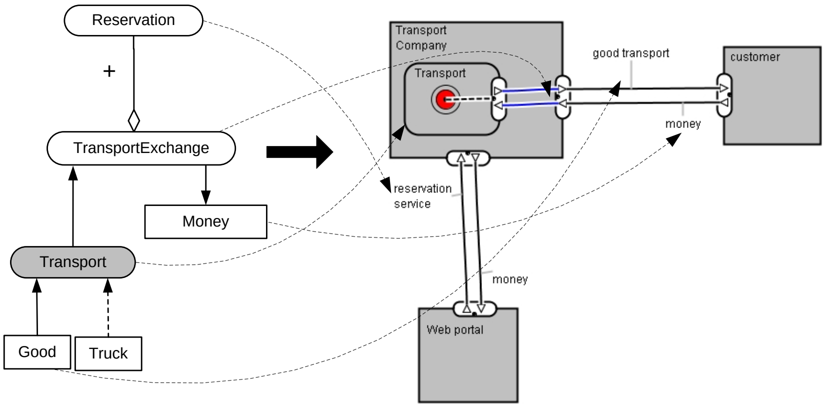

Figure 3-6: Value network perspective

3.5.3 The Business Process Perspective

First, we describe the links between data model and business process and the e^3-value network and the business processes. Business process model and data model play important roles in information system construction. They represent two different perspectives of business knowledge. The data model describes the data that flows through the business processes. In business process diagrams, the data is represented as data stores. However, all the details represented by the ERD are not visible in the process model. For example: relationships among entities. Data stores of the process model cannot be derived, when mapping BSRM coordination view to the process model. Hence, the traceability between ERD and process model helps to discover the possible data stores in the process model.

Generating process models from more abstract models has been tried before. Several researchers (Weigand et al., 2007; Pijpers and Gordijn, 2007; Andersson et al., 2006; Wieringa et al., 2008) proposed to build the process model from an e^3-value model. Weigand et al. (2007) proposed a transformation technique starting from value model show how the ontological gap between the value model and the aspect models can be bridged using at least three process aspects: the resource management, the communication and the risk assessment. Pijpers and Gordijn, (2007) proposed a method that makes an intermediate model called e^3 transition model based on value model. Andersson et al. (2006) proposed a method that starts with value model and follows a chain of steps and identifies sub-processes. From this work, we can conclude that generating a process model cannot be formalized completely, as it involves making design choices. In this section we consider the mapping between the BSRM and the process model. We also make use of an intermediate step and aim at providing essential mapping rules rather than a formalized transformation.

3.5.3.1 Model View Points

In this particular mapping we introduce coordination view point for the BSRM model. So, first we describe the *coordination viewpoint* and then we move to the mapping between BSRM and process model. A viewpoint is an abstraction view of a system which is focusing on a particular set of concerns while suppressing all irrelevant details. Some of the key concepts become clearer and visible with different viewpoints. To have a better insight into BSRM, we distinguish two viewpoints namely Business Viewpoint and Coordination Viewpoint.

Coordination services operate on intentional resources. They correspond to operational services in the EM-BrA2CE ontology (Goedertier et al., 2007) that are said to change the state of business activities. They relate to both the production world and coordination world. For example, a Sales Order Management service supports a (production world) product exchange, but it is itself a coordination world conversion service that produces a commitment in the form of a Sales Order. When service models grow, it is possible for practical reasons to split up a BSRM model into a business service viewpoint and a coordination service viewpoint. The former focuses on the production world and leaves out coordination services, whereas the latter focuses on the coordination services and models them completely (the intentional resource types that they manipulate, the production world service that they support, but not the rest of the business service model).

Since intentional resource types are not material and typically need representation by means of informational objects, coordination services correspond closely to web services at the PIM level (cf. The Business Process Perspective section). We view them as boundary objects (Wenger, 1998) that support crossing the boundaries between the business domain and the IT system domain.

3.5.3.2 BSRM and Business Process Mapping

Business process modeling aims to describe the internal and external business processes of an organization. One widely-used notation in business process modeling is BPMN 1.1 (BPMN OMG, 2008).

The coordination viewpoint of the service model plays a pivotal role when mapping with the process model. This mapping can be used to generate an initial process model after the service model has been fixed. In other words, this mapping gives a guidance to identify basic core processes in the process level. These processes can be realized in different ways. So the BSRM model can be considered as an abstraction layer on top of the processes up to certain extend. To show how, we concentrate on the *ProductExchange* service (Figure 3-7). It corresponds to a swim lane in the process model. The coordination services map to sub-processes (that will often have the transaction property) in the swim lane, and resource dependencies between them map to a control flow relationship. Intentional Resources correspond to messages that are exchanged between processes. Strictly speaking, an Intentional Resource, e.g., a reservation, is an abstract entity, not a message, but just because it is abstract, it needs a data representation, which motivates this mapping.

In BPMN 2.0 exchange services would correspond to choreographies, but in BPMN 1.0 we have to split them up over two swim lanes. Within BPMN, sub-processes can be decomposed into a series of activities and message exchanges. If we define coordination services strictly on the basis of coordination objects, each service sub-process has a well-defined number of possible activities and messages (like “request”, “cancel”), besides the coordination object itself. To derive the activities and messages, one can draw on well-established transaction patterns (Dietz, 2006; Weigand and van den Heuvel, 1999).

The value network model provides an additional input to the process model: actors in the value network correspond to pools in the process model. Data stores are derived from the ER model.

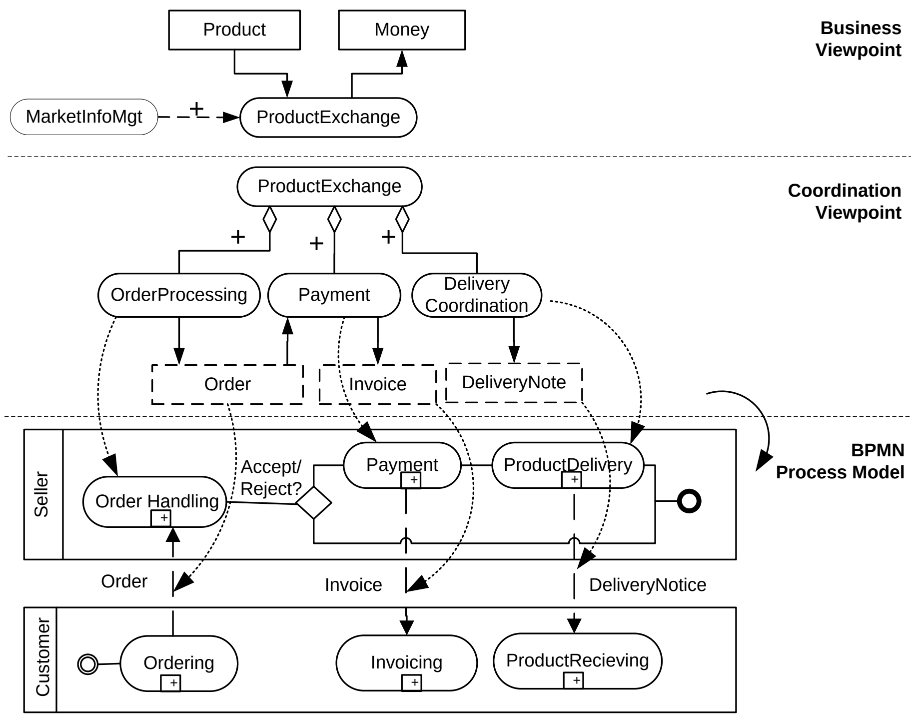

Figure 3-7: Business process perspective

In principle, the mapping is straight-forward. Coordination services correspond to basic web services whereas exchange services correspond to “business services” in the sense of (Papazoglou and van den Heuvel, 2006), that is, orchestrations of web services. It is possible to formalize the service model mapping with BPMN process model by deriving mapping rules as listed below.

R1: Every exchange service in the service model corresponds to at least two swimlanes in the process model.

R2: For every coordination service in the service model, there is a sub-process in a swimlane in the process model.

R3: Core sub-services in both exchange and conversion services correspond to sub-processes as well. The control flow must be aligned with the resource dependencies.

R4: For every intentional resource in the service model, there is a message exchange in the process model.

3.6 BSRM Viability Check

Evaluation and validation of research artifacts is part of the design science approach. The accuracy of the modeling language is an important indicator of its viability. As BSRM is a new modeling language, the systemic evaluation and validation of the modeling technique is of vital importance. The validation of the research is discussed in chapter 6. Part of the evaluation is covered in this section. The literature review of evaluation techniques for systems analysis and design done by Siau & Rossi (2008), shows that the metamodeling approach is one of a common model evaluation technique. Metamodeling represents a modeling process that takes place at one level of abstraction and logic higher than the standard modeling process (Gigch, 1991). As we derive the constructs and logic of BSRM from the metamodel, the model can be evaluated in an objective way. The concepts and the constraints in the metamodel were defined using ConceptBase which provides support to any OMG MOF abstraction level. In particular metamodel implemented with ConceptBase represents M2 level. When deriving the instance models the constraints have to be satisfied. As the service metamodel is defined as classes, attributes, associations and constraints in ConceptBase, the correct use of above modeling concepts and logic at model level is automatically guided.

Secondly, the BSRM feature comparison with other service modeling approaches (which were discussed in the background section) is also included as a part of evaluation of the proposed modeling language. As we followed MDA approach and business modeling perspective, these two were selected as features for comparison. “Horizontal and vertical mapping with other models” are considered as important features, because it reveals how much constructive the new model with other well established models. As resources are important ingredient of value co-creation which is the main focus of the service, we considered the resource-service relationship as the next aspect to be compared. Finally modeling notation is also taken into account. The results are listed in Table 3-2.

Table 3-2: Feature Comparison of BSRM with other service modeling approaches

Features	BSRM	SOMA	SOD-M	WSMO	SOAML	Weigand,et al., (2009)
Correspondence with MDA levels	CIM, PIM, PSM	CIM, PIM, PSM	CIM, PIM, PSM	PIM, PSM	CIM, PIM, PSM	CIM, PIM, PSM
Horizontal Mapping with other models	(Partial) e^3-value model and ER model		(Partial) e^3- value model and business process model		(Partial)	e^3-value model
Vertical Mapping with other model	(Partial) BPMN		(Partial) Service process model using BPMN	(Partial) used UML class descriptions.	(Partial) BPDM	e^3-value model to map vertical mapping
Basis for the Business Service Model	REA	Goal service modeling	e^3-value model and business process model provides list of business services	User desired goals are captured and listed down in WSML language.	Business motivation model	e^3-value model
Classification of Services	Yes	No	No	No	No	Yes
Consideration of resources – service relationship	Yes	No	No	No	No	No
*Focus is on BS / SS / Both	Both Provides BS modeling language and notation	SS	Both Provides text based list of BSs	SS	SS	Both Provides text based list of BSs
Modeling notation	Own notation and UML	UML	e^3 -value and UML	UML, Meta modeling approach	UML	e^3-value notation

*BS –Business Services, SS - Software Services

Feature comparison of BSRM with other approaches reveals several similarities as well as specific features to BSRM. However, none of the above research works provides a systematic notation of service design at CIM level except BSRM. Although the e^3-value business model is rich enough to identify value exchanges in a network, value co-creation is not addressed well. As the resources are first class citizen of REA, BSRM gives a better insight to the value co-creation which is the main focus of services. All in all, we can conclude that a major feature of BSRM is strong business thinking.

3.7 Service Model Analysis

Although the value of CIM business models for system development is clear, business analysts are often not very motivated to cooperate if the models do not support some analysis as well. Indirectly, such an analysis also benefits the developer, as it will improve the quality and stability of the models. Although, we mention several analysis for business users, the complete analysis method is not given for the first two points. Without going into details, we mention the following possibilities.

- *Value chain analysis*
  Value chains (in BSRM built up from service/resource links) have been used in audit theory for control purposes, but are also a basis for cost analysis and the analysis of entrepreneurial viability (Griffioen et al., 2000). On the other hand, the requirement list for model- based auditing (Weigand and Elsasb, 2012) shows that REA which is the underlying business ontology of the BSRM, aligns with the fundamental auditing principles. Linkages between services (across the boundaries of business units and the organization) should be analyzed, for instance, for the synergy of merging similar services into generic shared ones (Porter, 1985; Stabell and Fjeldstad, 1998). An advantage of BSRM is that it can model both the primary value activities and the supporting value activities in a formal way. This in contrast to most process models, that only focus on the primary activities.

- *Customer value analysis.*
  Following SD-logic (Alter, 2008; Grönroos, 2008), the BSRM model can be extended to the customer or business partner domain, to analyze where and how customer value is created. For example, a transportation service creates value for a business customer as it supports its logistic process. Customer value analysis may proceed by modeling this logistic process, and improvements could be found, either by extending the transportation service or by improving the coordination with other services in the logistic process, for instance, customs services.

Chapter 4

Business Service Patterns

Patterns are used in many areas for different purposes. The patterns in software design are reusable objects which reduce the design time while providing the template of domain concepts. The analysis pattern book by Martin Fowler (1996) is a good example. The systematic use of patterns promotes the quality, standardization, reusability and the maintainability of the software artifacts (Bottoni et al., 2010). In this chapter we describe the Business Service Patterns (BSPs), its operations and its usage in model composition. Business patterns are reusable models that describe how the companies perform businesses (Hruby, 2006). Business Service Patterns describe the structure of the business from a service perspective. The purpose of using business service patterns is twofold. Firstly, the reuse of patterns reduces the design time while assuring that the designer does not violate the domain concepts. Secondly, it provides guidance to capture the real business activities as services.

All the patterns presented in this dissertation are based on pattern structure specifications. The pattern structure specification consists of following details. It provides the graphical structure, the basic information and the constraints to construct the pattern. As it is not possible to provide all the business service patterns for a particular business domain within this research, we limit ourselves to the most fundamental business activities. Hence, we select the value chain activities of Porter's (Porter, 1985) for manufacturing domain and several selected area in service domain. We develop minimal set of BSPs, within each category of Porter's value activities. For the service domain, we develop one generic pattern for all kinds of service industries and for selected areas including utility services, personal care, insurance, etc. As we mentioned in chapter 2, Hruby (2006) defines more than 20 business patterns based on REA that describes the structure and the functionality of a business. Therefore, we select the same examples as in Hruby (2006), wherever possible.

As patterns are used to symbolize repetitive activities, they represent more standardized businesses occurrences. However, today's companies constantly search for new business opportunities to satisfy the customer's diverse needs and to survive in the business. Prahalad and Krishnan (2008) addresses the above dual aspects by defining a new landscape of business, driven by consumer co-creation and service customization. The main focus of their work is serving one customer at a time (formulated as N=1) and sourcing resources globally (formulated as R=G). The firms should build capacities to access the global network of resources to co-create unique experiences with customers. Hence, the firm requires internal capacity to reconfigure resources in real time. So, the standard business solutions are not sufficient to deal the dynamic business needs. The proposed service design framework is flexible enough for the expansions and the modifications in systematic way. The merging of patterns is defined through Business Service Pattern Operator called "*Merge*". Moreover, the

Outsourcing Pattern Specification provides an opportunity to model the concept of global sourcing of resources - R=G (Prahalad and Krishna, 2008).

The patterns alone do not create a model. The systematic composition of patterns creates a model. The research work of Bottoni et al. (2010) which is grounded in category theory (Lane, 1998) provides a formal approach for pattern specification and pattern-based model completion. We follow the definitions suggested by Bottoni et al. (2010), for pattern structure specification and pattern composition. We present two operations for the pattern structure specifications to derive patterns. The pattern composition is described using Business Service Pattern Operators (BSPOs). Finally, we provide the design steps to guide the designer to construct the business service model for the given problem.

Section 4.1 of this chapter defines the pattern structure specifications. There are seven pattern structure specifications which represent the five categories of services identified in the service metamodel. These specifications describe the basic information about the service which they represent. Following Bottoni et al. (2010), we define pattern operations to derive domain specific patterns using the pattern structures in section 4.2. Porter's value chain activities and it's alignment with business services are described in section 4.3. We present generic BSPs for each category of primary and supporting activities in sections 4.4 and 4.5 respectively. The BSPs for service industries are described in section 4.6. The section 4.7 describes the business service pattern operators. Section 4.8 provides design steps to construct the enterprise business service model. The chapter ends with providing service integration mechanism with web services.

4.1 Specification for Business Service Pattern Structures

A metamodel offers the concepts for formulating a valid model. The exact relationships for a given concepts are not explicitly visible in the metamodel. For example the metamodel shows the relationships of the exchange service and the resources as stock inflow and outflow. But this relationship has to be further described as give and take relationship in the model level. Hence, we define pattern structures on top of the definitions in the metamodel. We define seven specifications for business service pattern structures for five categories of services (exchange, conversion, sub-service, coordinate and enhance) identified in the service metamodel. Patterns are derived using one or more of these five categories. Therefore, all the patterns follow the pattern structure specification/s which they are built.

The pattern structure specifications are based on a formal background. We follow Bottoni et al. (2010), which proposed a formal approach to the specification of patterns, pattern analysis and model completion. Following Bottoni et al. (2010), we identify the pattern structure as a *variable* pattern. The variable pattern contains a fixed part called *root*, and a variable partV_i. The variable part can be replicated.

Definition : *A variable pattern is defined as*

VP=(P, root, Emb, name, var)

Where

- $P= \{V_1,.....,V_n\}$ is a finite set of non-empty graphs where each V_i is called variable part,

- *root* is a distinguished element of P, also called the fixed part,
- Emb is a set of morphisms $v_{i,j}: V_i \rightarrow V_j$ with $V_i, V_j \in P$, such that it spans a tree rooted in $root \in P$ with all graphs $V_i \in P$ as nodes and morphisms $v_{i,j} \in Emb$ as edges,
- *name:* $P \rightarrow L$ is an injective function assigning each variable part a *name* from a set of variables L, of sort ℕ
- $var \subseteq T_{AlgEq}(name(P))$ is a set of equations governing the number of possible instantiations of the variable parts. These equations use variables in $name(P) \subseteq L$, arithmetic operations and are restricted to use the <, ≤, =, >, ≥ relation symbols. We call this signature "*Algebraic Inequalities*" ($\sum_{AlgEq)}$ and hence$T_{AlgEq}(name(P))$ is the term-algebra with variable in *name (P).*

For example, the Figure 4-1 shows the pattern structure of the *exchange* service. For the exchange service pattern structure, exchange service is the fixed part (the root), and the other services and resources can vary. The valid expansions and the minimal requirements of the variable part are defined in the constraints of the pattern structure. (detailed specification of exchange service pattern structure is in section 4.2).

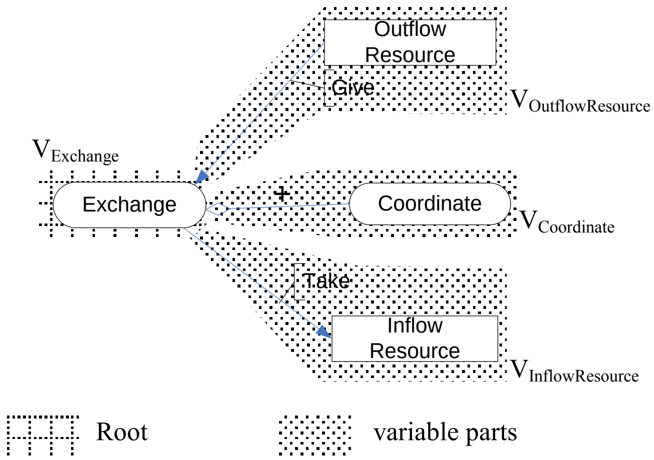

Figure 4-1: Variable Pattern

Here we present the full definition with the example.

VP= (P=($V_{Exchange}$, $V_{OutflowResource}$, $V_{Coordinate,}$ $V_{InflowResource}$) ,

root= $V_{Exchange}$,

Emb= ($v_{Exchange, OutflowResource}$: $V_{Exchange} \rightarrow V_{OutflowResource,}$

$v_{Exchange, Coordinate}$: $V_{Exchange} \rightarrow V_{Coordinate}$.

$v_{Exchange, InflowResource}$: $V_{Exchange} \rightarrow V_{InflowResource}$),

name= {($V_{Exchange}$, Exchange), ($V_{OutflowResource}$, OutflowResource), ($V_{Coordinate,}$ Coordinate), ($V_{InflowResource,}$ InflowResource)},

Var=[OutflowResource ≥ 1, Coordinate≥ 0, InflowResource≥1])

Each morphism ($v_{Exchange,\ OutflowResource}$, $v_{Exchange,\ Coordinate}$, $v_{Exchange,\ InflowResource}$) in the embedding set represents the mapping from Exchange service to OutputResource, Coordinate and InputResource accordingly. Each part of the *Var* describes the constraints when instantiating the variable part.

OutflowResource ≥ 1: at least one outflow resource is required in the Exchange service pattern

Coordinate≥ 0: coordination service is always not required in the Exchange service pattern

InflowResource≥ 1: at least one inflow resource is required in the Exchange service pattern

The business service pattern structure specification describes the structure of the service, resource and their relationships in a general form and the constraints which has to be followed. In the specification there are no domain specific services or resources. We use the following style to document the pattern structure specification.

Name: Describes the name of the specification of business service pattern structure. We select the first three letters of each name. (when there are sub categories under one specification, we use three letter plus another letter as a suffix)

Description: Describes the basic information about the business service, which is included in the structure.

Parameters: Describes the parameters.

Pattern: Describe the pattern

Constraints: Describes the minimal requirements that has to be satisfied by the pattern structure and the restrictions

Graphical Structure: Describes the graphical representation of the pattern structure.

4.1.1 Specification of the Exchange Service Pattern Structure

The exchange service is one specialization of the service type in the service metamodel. According to the REA ontology, exchange corresponds to a group of decrement and increment economic events. In this section, we describe the specification for exchange service pattern structure. Depending on the constraints we distinguish three categories of *exchange* service. The first one is exchange of physical/intentional resource. The exchange of service appears in two ways. i.e. sale of service and outsourcing. We demonstrate these three categories in the next sub sections.

4.1.1.1 Specification of the Exchange Service Pattern Structure (Exchange of Resource- Physical or Intentional)

Name: Exchange_Resource

Description: "*Exchange*" is a giving something valuable to another party in return of another valuable thing. This is a basic business transaction in any company. It can occur business to business, business to customer, business to supplier, business to government etc. The exchangeable thing is a physical resource or intentional resource.

Parameters: *FA*: Focal Agent, *EA*: External Agent

Pattern: *Exchange service* has *give* and *take* relationship with resources. The business service pattern structure for exchange of physical resource service is shown in Figure 4-2.

Constraints:

- At least one resource inflow and outflow have to be defined for the exchange service
- At least one resource inflow of the exchange service belongs to the focal agent. This ensures that the focal agent really contributes something
- At least one resource outflow belongs to the external agent. This ensures that the service has value for the external agent
- Enhance, coordination and core-sub services are optional

The constraints are represented as ConceptBase query class (Staudt et al, 1993; Staudt et al, 1994; Jeusfeld, 2008) as given below. The ResourceExchange service is defined as a query class with two parameters (focal agent and external agent). The constraint name is c1 and it is defined within the $ symbol. The word "this" refers to the query class – ResourceExchange.

```
ResourceExchange in GenericQueryClass isA ServiceType with
parameter
       FA: AgentType;
       EA: AgentType
constraint
c1: $ exists OutflowResource,InflowResource/EconomicResourceType
            (this give OutflowResource) and
            (this take InflowResource) and
            (FA control OutflowResource) and
            (EA control InflowResource) and
            (EA \= FA)
     $
 end
```

Graphical Structure:

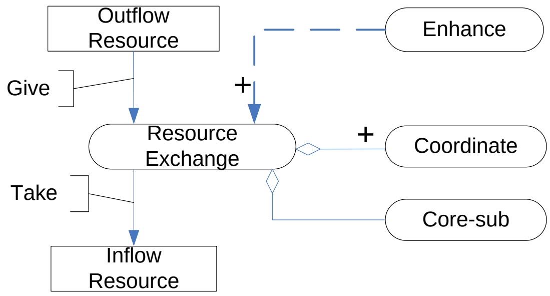

Figure 4-2: Structure of the Exchange Service Pattern- (Exchange of physical resource)

4.1.1.2 Specification of the Exchange Service Pattern Structure (Sale of a Service)

Name: Exchange_Service

Description: Sale of service is one way of "*Exchange*" service. The company provides some service which is valuable to another party in return of another valuable thing. The fundamental difference between sale of a physical resource and a sale of service is the resources attached to the selling service.

Parameters: *FA*: Focal Agent, *EA*: External Agent

Pattern: *Sale of service* has *give* and *take* relationship with a service and resources respectively. The service which is exchanged always affects a resource of the external agent. The business service pattern structure for the Sale of service is shown in Figure 4-3.

Constraints:

- At least one service outflow and one resource inflow have to be defined for the exchange service.
- At least one of the resource inflow of the exchanging service(*s*) belongs to the focal agent. This ensures that the focal agent really contributes something.
- At least one resource outflow of the exchanging service (*s*) belongs to the external agent. This ensures that the service has value for the external agent.
- Enhance, coordination and core-sub services are optional

The constraints are represented as ConceptBase query class as given below.

Specification for Business Service Pattern Structures

```
SaleOfService in GenericQueryClass  isA ServiceType  with
parameter
        FA:AgentType;
        EA:AgentType
constraint
c2: $ exists Service/ServiceType InflowResource2/ResourceType
      (this give Service) and
      (this take InflowResource2)
and
      (exists OutflowResource/ResourceType
      (Service use OutflowResource) and
      (FA control OutflowResource))
and
      (exists InflowResource1/ResourceType
      (Service produce InflowResource1) and
      (EA control InflowResource1))
and
      (EA \= FA)
    $
end
```

Graphical Structure:

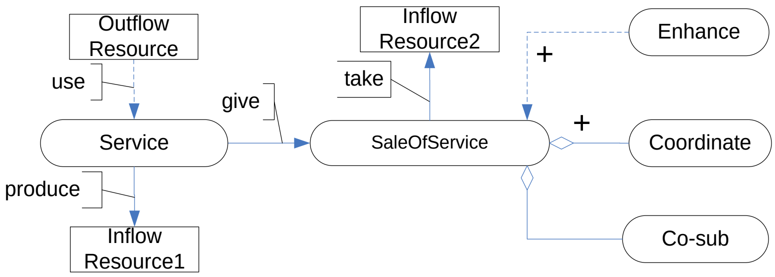

Figure 4-3: Structure of the Sale of Service Pattern

4.1.1.3 Specification of the Outsourcing Service Pattern Structure

Outsourcing is a practice used by companies to reduce costs by transferring portions of work to outside suppliers rather than completing it internally. According to Grossman "We live in an age of outsourcing. Some firms become *virtual* manufacturers, owning designs for many products but making almost nothing themselves" (Grossman, 2005). Outsourcing has been practiced ages and it has many advantages as well as disadvantages. One major advantage is better control over internal cost (Prahalad and Krishna, 2008). There are several advantages and disadvantages of outsourcing. Outsourcing allows to create a greater focus on core business and acquire expertise knowledge and experience. It improves the productivity. Some of the disadvantages are less control over the outsourced service, security issue, and staff turnovers. In this section, we demonstrate the outsourcing pattern structure.

Name: *Exchange_Outsource*

Description: Outsourcing service pattern structure follows the exchange service concepts. The fundamental difference between sale of a service and outsourcing a service is the control over the resources attached to the sourcing service. The focal agent has to give something valuable to acquire a service from external party. The service which is outsourced always uses some resources which belong to the external agent and the outsourced service affect at least to one resource/service of the focal agent.

Parameters: *FA*: Focal Agent, *EA*: External Agent

Pattern: Outsourcing service pattern structure has *give* and *take* relationships between *ServiceOutsourcing* and the *money* and the *service*, respectively. The service which is outsourced has stock inflow and outflow relationships with resources. In particular, it *uses* resources (OutflowResource) and it *produces* another resource (InflowResource) - the meaning of "produce" can be actual production or adding value. The business service pattern structure for outsourcing service is shown in Figure 4-4.

Constraints:

- At least one resource inflow and service outflow have to be defined for the exchange service
- At least one of the resource outflow of the outsourced service (os) belongs to the focal agent. This ensures that the service has value for the focal agent.
- At least one resource inflow of the outsourced service (s) belongs to the external agent. This ensures that the external agent really contributes something.
- Enhance, coordination and core-sub services are optional

The constraints are explained in English below.

The constraints are represented as ConceptBase query class as given below.

```
OutsourcingService in GenericQueryClass  isA ServiceType  with
parameter
       FA:AgentType
       EA:AgentType
constraint
c3: $ exists Service/ServiceType OutflowResource2/ResourceType
      (this give OutflowResource2) and
      (this take Service)
and
     (exists OutflowResource1/ResourceType
     (Service use OutflowResource1) and
     (EA control OutflowResource1))
and
     (exists InflowResource/ResourceType
     (Service produce InflowResource) and
     (FA control InflowResource))
and
     (EA \= FA) $ end
```

Specification for Business Service Pattern Structures

Graphical Structure:

The graphical structure is very similar to a sale of service. There are two differences. The resource inflow and outflow for the exchange service are defined in opposite way. The Sale of Service is replaced by Outsourcing service.

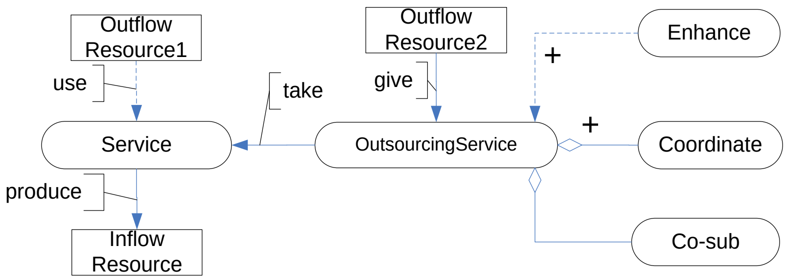

Figure 4-4: Structure of the Outsourcing Service Pattern

4.1.2 Specification of the Conversion Service Pattern Structure

According to REA, conversion corresponds to a group of decrement and increment economic events. As a result of conversion, a new product / service can be created or features of existing product / service can be changed. Several examples for conversion are producing, baking, cultivating, excavating and extracting. We distinguish conversion as a service type. One reason to define the conversion as a service is view the business as a service business and constructs the service model. As a result of separating conversion as a service, it can be related to an interface. Another advantage of modeling conversion as a separate service is the flexibility of outsourcing that particular service. In this section, we describe the specification for conversion service pattern structure.

Name: Conversion

Description: The simple definition of "conversion" is transform input into output. The production process is good example for the conversion.

Pattern: The conversion service has at least one *use* or *consume* relationship with resources. The conversion service *produces* at least one resource/service. The *produce* means creating or adding value. The business service pattern structure for conversion service is shown in Figure 4-5. According to the BSRM notation, the conversion service is depicted with colored rounded rectangle.

Constraint:

- At least one resource inflow and two outflow has to be define for the conversion service
- Enhance, coordination and core-sub services are optional

The constraints are represented as ConceptBase query class as given below.

```
Conversion in GenericQueryClass  isA ServiceType  with
constraint
c4: $ exists OutflowResource1, OutflowResource2, InflowResource/
      ResourceType
        (this consume OutflowResource1) and
        (this use OutflowResource2) and
        (this produce InflowResource)
    $
end
```

Graphical Structure:

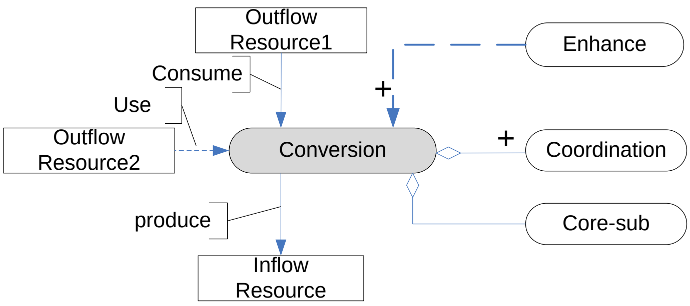

Figure 4-5: Structure of the conversion service pattern

4.1.3 Specification of the Sub-service Pattern Structure

The next category of service is sub-service. Sub-services have to be introduced when a realization of a service involves multiple value activities and it makes sense (economically) to view these value activities as independent services that are shared by different contexts. Moreover, decomposing a service to sub-services helps to spread certain fixed costs over a larger number of activities. As we described in the service metamodel, there are two categories of sub-services namely core-sub service and coordination service. In this section, the specification is defined for the core-sub service. The coordination service is described in the next section.

Name	*:* Sub-Service
Description	*:* Sub-services are coming into play when the core service has to be realized by multiple activities. For example produce is a core service and it has assemble, and inspection sub-activities. When decomposing the composite service, it is important to analyze the relationship between resources with the sub-activities and their constraints.

Specification for Business Service Pattern Structures

Pattern : Sub-services have a part-of relationship with the composite service. The business service pattern structure for sub-service is shown in Figure 4-6.

Constraints :

- At least two sub-services are to be defined.
  Note: Number of sub-services can be more than 2. As pattern structure is a variable pattern, the number of sub-services can be increased. The pattern structure has a fixed part and a variable part; we consider the sub-services and its related resources as the variable part.
- Inflow resources must be consumed by a sub-service, outflow resources must be produced by a sub-service.
- Intermediate outflow resources have to be consumed by another sub-service.
- Intermediate inflow resources have to be produced by another sub-service.

The constraints are represented as ConceptBase query class as given below.

```
CompositeService in GenericQueryClass  isA ServiceType  with
constraint
c5: $ exists SubService1, SubService2 /ServiceType
      (this partOf SubService1) and (this partOf SubService2)
and
      (exists OutflowResource, InflowResource/ResourceType
      (SubService1 consume OutflowResource) and
      (SubService2 produce InflowResource))
and
      (exists IntermediateResource/ResourceType
      (SubService1 produce IntermediateResource) and
      (SubService2 consume IntermediateResource))
and
      (SubService1 \= SubService2) $
end
```

Graphical Structure:

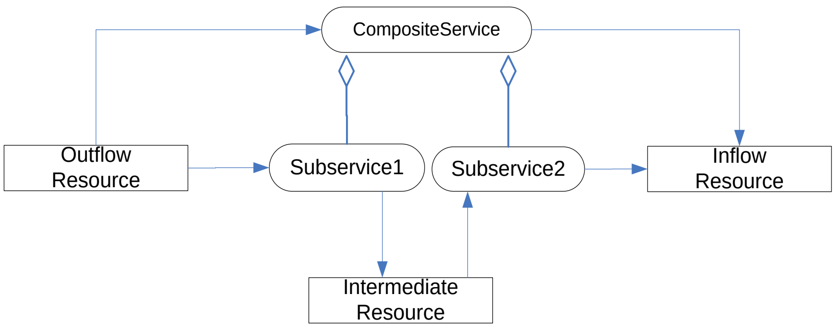

Figure 4-6: Structure of the conversion service pattern

4.1.4 Specification of the Coordination Service Pattern Structure

The coordination service is a sub-service, which coordinates the dependencies of multiple sub-services. Further, it can be seen as enhance sub-service, because it adds value to the sub-services. The coordination service manipulates intentional resources.

Name : Coordinate

Description : The coordination service is defined as an enhance sub-service: if B coordinates A, then B enhances A (A is the goal of B) and B is used somewhere in the realization of A. It manipulates intentional resources.

Pattern : The business service pattern for coordination service is shown in Figure 4.7. The coordination service has a coordinate relationship with the service which is coordinated.

Constraints :

- Coordination service coordinates at least one service.
- Coordination service can have sub-services or enhance services. Those are optional,
- Intentional resources are optional

The constraints are represented as ConceptBase query class as given below.

```
CoordinationService in GenericQueryClass  isA ServiceType  with
constraint
c5: $   exists Service /ServiceType
        (this coordinate Service)
     $
end
```

Graphical Structure:

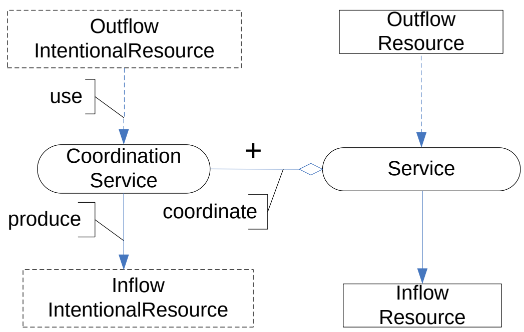

Figure 4-7: Structure of the coordination service pattern

4.1.5 Specification of the Enhance Service Pattern Structure

Enhance service adds value to other service, by improving effectiveness or efficiency. It has a goal of another service. It cannot be exist interdependently. Enhance services are explained in the chapter 5.The pattern structure is same as the Coordination pattern structure.

Name : Enhance

Description : A service that adds value to the any other service called its goal, are considered as enhance service. Assume that service A has a "goal" of service B. In terms of REA, enhance corresponds to a stock-outflow ("produce") relationship between two services, for instance, a management service and an operational service. Enhance service manipulates intentional resources.

Pattern : The business service pattern for the enhance service is shown in Figure 4-8. The enhance service has a enhance relationship with the service which is enhanced. It has *use* and *produce* relationship with intentional resources. But this requirement is not mandatory.

Constraints:

- Enhance service enhances at least one service.
  Enhance service can have sub-services. Those are optional
- Intentional resources are optional

The constraints are represented as ConceptBase query class as given below.

```
EnhanceService in GenericQueryClass  isA ServiceType  with
constraint
c5: $  exists Service /ServiceType
       (this enhance Service) $
end
```

Graphical Structure:

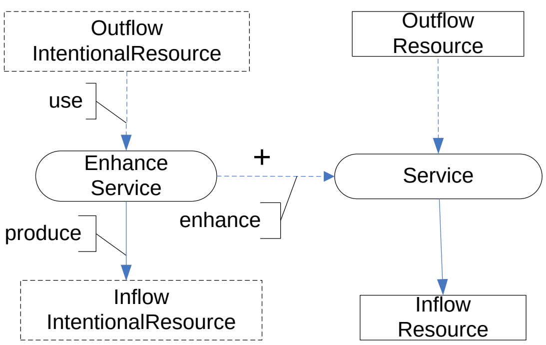

Figure 4-8: Structure of the Enhance service pattern

4.2 Operations on Pattern Structures

Pattern structures are the skeletons of the patterns and models. Pattern structures provide abstract view of the service and its relationships with other resources/services. Patterns and pattern based models can be derived using structures defined in the previous section. To use the pattern structures in constructive way for modeling, it is necessary to describe a formal approach for its operations. Based on the pattern expansion and pattern annotation definitions Bottoni et al. (2010), we define two operations for expansion and the annotation of the pattern structure.

4.2.1 Expansion of Pattern Structures

The pattern structures described in the previous section are variable patterns. As the pattern structure has a variable part, it can be expanded within the given constraints. According to Bottoni et al. (2010), we denote pattern expansion as *EXP (VP)* and it is called expansion set. The expansion is not limited to the pattern structures. It can be applied to the patterns as well. Following the definition of pattern expansion given in Bottoni et al. (2010), we describe the pattern expansion.

Definition:

> *Given a variable part (VP),its expansion set EXP(VP) is given by all graphs E_{ij} such that, there is a subjective function $f_{ij}: C_i \rightarrow E_{ij}$ from the set of all colimits $[C_i]$of all possible diagrams α obtained by replicating graphs in P, and the morphisms in Emb, such that:*
> *- the diagram α is consistent with the morphisms in Emb, which means that if $V_i \rightarrow V_j$ is included in α, then there is a morphism $v_{ij}: V_i \rightarrow V_j$in Emb*
> *- the number of replicas in each path from root to C_i satisfies the equations in Var.*

Example:

To demonstrate the pattern structure expansion, we select the Sub-Service (sub-service pattern structure) specification. Figure 4-9 (a) shows the sub-service (variable part) pattern structure with minimal number of sub-services. According to the Sub-Service specification constraint, the variable part can be expanded. Figure 4-9 (b) shows a valid expansion by adding another sub-service and a resource.

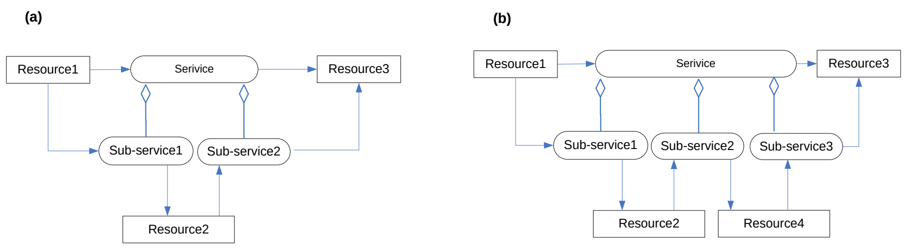

Figure 4-9: Pattern structure expansion

4.2.2 Annotation of Pattern Structures

The second operation is *annotation*. In order to derive a domain specific pattern from the pattern structure specification, we use the pattern annotation operation. The annotation mechanism is ground on the notion of triple graph (Guerra and de Lara, 2007), (*source* and *target*, related through a *correspondence* graph). The definition is given below. For pattern annotation, we need its structure (as given by a variable pattern), a vocabulary of pattern roles, and a mapping from the elements in the pattern structure to the vocabulary.

Definition: *Pattern annotation define as a triple graph*

> *A triple graph* ($G_s \overset{c_s}{\leftarrow} G_c \overset{c_t}{\rightarrow} G_t$) *has three graphs* G_i ($i \in \{s, c, t\}$), *and two functions* $c_j : V_{G_c} \rightarrow V_{G_j}$. *A node or edge x of* G_s *is related to a node or edge y of* G_t, *iff* $\exists n \in V_{G_c}$ *s.t.* $x \overset{c_s}{\leftarrow} n \overset{c_t}{\rightarrow} y$

Example:

The annotation operation is illustrated using a simple example of bike producing. Figure 4-10 shows the pattern annotation for bike producing as a triple graph. In this example we consider bike producing is accomplished by assembling and painting activities. First, frame and the wheels are assembled together and then the assembled bike is painted. To model this example with BSRM, we select the Sub-Service specification as the source graph- *Gs* (the lower part of the Figure 4-10). As the bike producing has two sub services, there is no need to expand the basic structure. The vocabulary consists of bike produce, wheel, frame, assemble, paint, assembled bike and the bicycle. The pattern vocabulary (the upper part of the Figure 4-10) represents the target graph- G_t. The matching between above two graphs is done by corresponding model- G_c (the middle part of the Figure 4-10). The morphisms from the nodes in G_c are shown as dotted lines .The result of the annotation is shown in the Figure 4-11. As the pattern structure is used as the source graph, the constraints relate to the structure remain in the resulting graph.

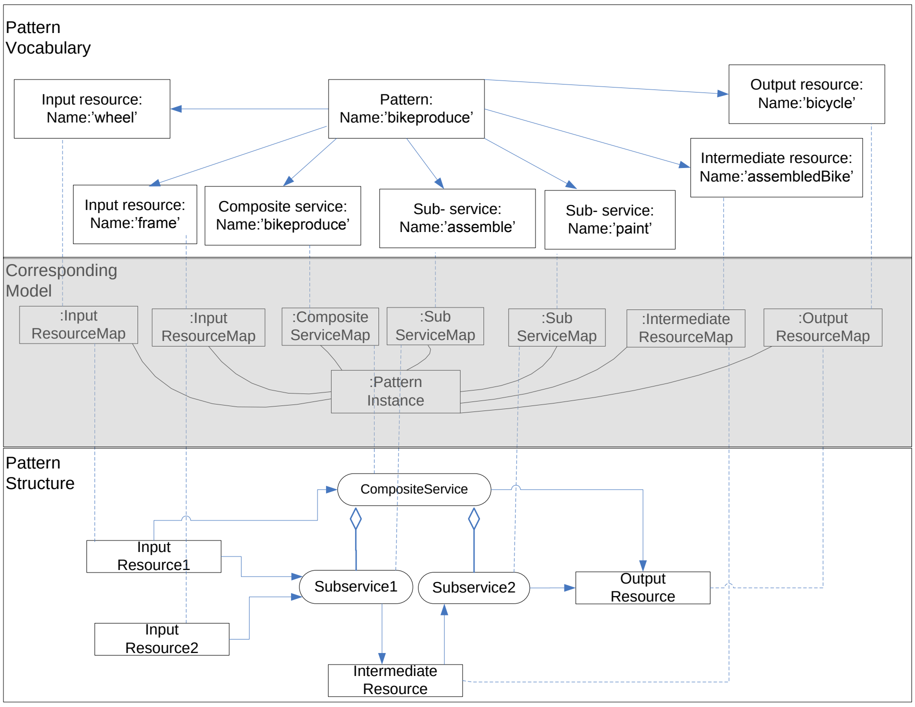

Figure 4-10: Pattern structure annotation

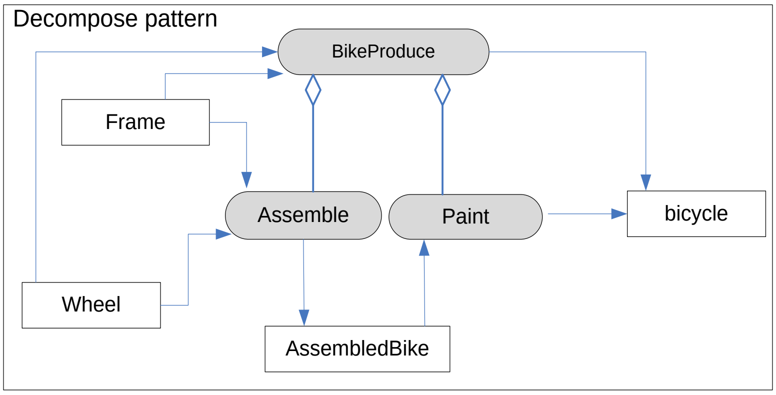

Figure 4-11: BSRM model for the bike producing

4.3 Business Service Alignment with Porter's Value Activities

In this dissertation, it is not possible to present business service patterns which cover all the activities in a particular domain. Hence, we set the boundaries using Porter's value chain activities (Porter, 1985) to demonstrate minimal set of business service patterns. Porter's value chain analysis describes the activities of a business which fall into primary and support categories, within and around an organization (Figure 4-12). He evaluated which value each particular activity adds to the organizations products or services. The primary activities are directly concerned with the creation and delivery of a product or service. The primary activities are grouped into five main areas: inbound logistics, operations, outbound logistics, marketing and sales, and service. Each of the primary activities is linked to support activities which help to improve their effectiveness or efficiency. There are four main areas of support activities: procurement, technology development (including R&D), human resource management, and infrastructure (systems for planning, finance, quality, information management etc.).

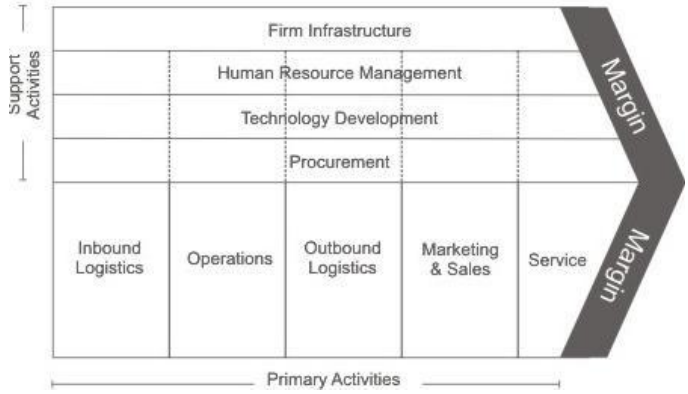

Figure 4-12: Porter's value chain model

We align the business services with Porter's value chain analysis. The Figure 4-13 represents the alignment of business services with Porter's value chain activities. According to Porter, primary activities are directly concerned with the creation or delivery of a product or service. These are core activities in a company. With regards to business service definitions, the conversion services represent the production and the exchange services represent the sales of good or service in return of something valuable. The conversion and exchange services become core services in practice. Therefore, the representation of primary activities in Porter's value chain and the representation of conversion and exchange services in business services are aligned together. Of course this alignment is not hundred percent. There are slight deviations in some points. For example marketing is a primary activity in Porter's value chain, but our view is it is not a core service. The deviations are denoted with * marks in the picture.

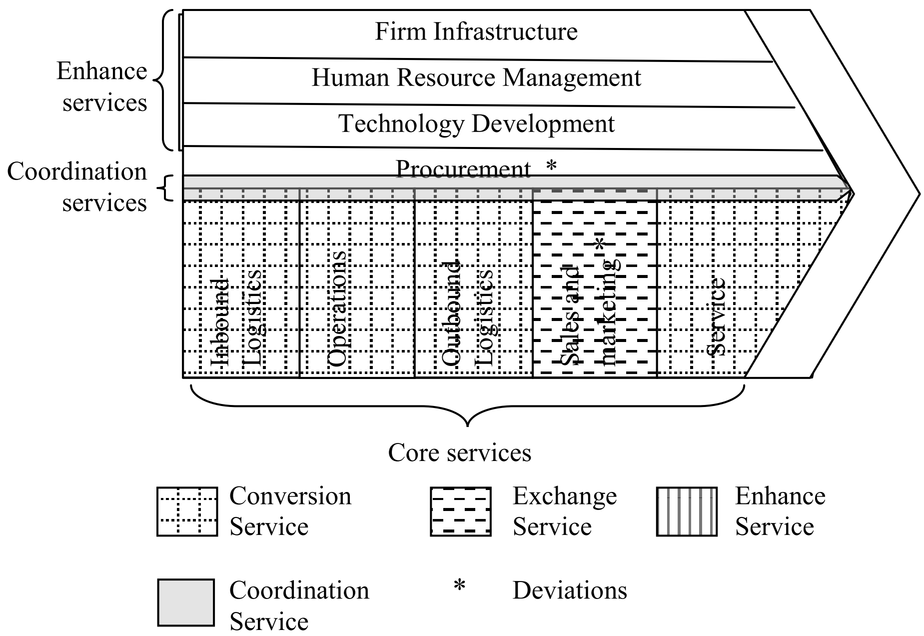

Figure 4-13: Alignment of business services with Porter's value chain activities

Each of the primary activities is linked to support activities which help to improve their effectiveness or efficiency. Similarly, enhance services help add value to any other service/product by improving efficiency and effectiveness. These two concepts lie together. Hence we distinguish that most of the enhance services are supporting activities. The detail discussion of enhance services is available in chapter 5. Again there is a deviation with procurement. According to Porter, procurement is a supporting activity. But we categorize the procurement as an exchange service, which is a core service. The role of coordination service is coordinating the primary activities and supporting activities. Therefore, the coordination services present in between both categories of the value activities.

We can observe similar approaches in business process point of view which is align with Porters value chain activities. There are three types of business processes: Operational processes, processes that constitute the core business and create the primary value stream. Typical operational processes are purchasing, manufacturing, advertising and marketing, and sales. The second category is supporting processes, which support the core processes. Examples include accounting, recruitment, call center, technical support. The third category is management processes, the processes that govern the operation of a system. Typical management processes include

"Corporate Governance" and "Strategic Management". The operational process describes the primary activities and the supporting and management processes together describe the supporting activities of Porters value chain.

In next section, we demonstrate one sample business service pattern for each primary and support activities of the value chain activities which are aligned with business services. Some more patterns which demonstrate the primary activities are available in Appendix B. The documentation of the business service patterns (BSP) has the following structure.

Pattern Name:	Name of the pattern
Description:	Describe the service which is to be demonstrated by the pattern.
Problem:	Describe the problem explaining the need of the pattern
Assumption:	Describe the assumptions which are taken when deriving the pattern.
Pattern Structure:	Describe the structure of the pattern
Solution :	Describe the solution
Reference:	List down the possible connection to other patterns.

The documentation of the patterns includes the above attributes and a graphical picture.

4.4 Generic Business Service Patterns for Primary Activities

In this section we demonstrate one business service pattern for each category of primary activity. We select some examples from Hruby (2006), considering the suitability to demonstrate the service pattern. According to Porter, there are five main areas of primary activities: inbound logistics, operations, outbound logistics, marketing and sales, and service. Note that, we do not model marketing under primary activities. We consider the primary activities are directly concerned with the creation or delivery of a product or service. Human resource is involving in many primary and support activities. At this stage we model the primary activities without human service provisioning for the simplicity. It is discussed under the support activities and in chapter 5 also. To derive business service patterns, we use the specifications of pattern structures together with operations of patterns structure which were introduced in the previous sections. All the patterns represent the type level.

4.4.1 Operational Activities

All the activities which require to transform inputs into outputs are considered as operational activities of a company. We selected the following example (cf. Hruby, 2006) - chapter 8), to demonstrate the business service pattern for operational activity.

Example 1: Creating a new product

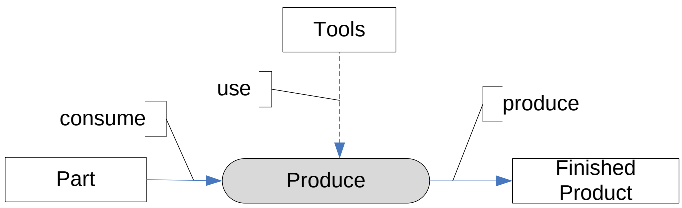

Figure 4-14: Business Service Pattern - Produce

Pattern Name:
BSP-Produce

Description:
Creating a new product or service is the core activity of many businesses. The production process converts input into a new form of output.

Problem:
How do we make a business service model for creating a new product?

Assumption:
We assume that the produce is an atomic activity. The production activity uses only the minimal resources.

Pattern Structure
The service model for creating a new product is based on Conversion specification (conversion service pattern structure- cf., 4.2.2). All the constraints of conversion pattern structure are to be satisfied when deriving the pattern for creating a new product. As the produce is considered as an atomic activity and it uses only the minimal resources in this example, the pattern structure annotation is sufficient to derive the pattern.

Solution:
Business service pattern for creating a new product is depicted in Figure 4-14. *Produce* is a conversion service. It *consumes Part* and it *uses Tool.* The stock outflow of the *Produce* service is *FinishedProduct*. The relationship between *Produce* service and the *FinishedProduct* is also *produce*. The waste produced by the production is not modeled here (refer BSP-Waste pattern). Business service model for producing of a new product is given below. Based on the assumptions, this pattern does not cover any intermediated stages (work-in process). All the intermediate stages are combined into a single conversion process.

Reference:
- *BSP-Produce* pattern can be connected with *BSP-Sale* pattern using the *merge* operator. The *FinishedProduct* is the input resource of *BSP-Cash_Sale*.
- *BSP-Produce* pattern can be connected with *BSP- Produce_Intermediate_Stages*, to illustrate intermediate stages of the production. The *part* is the input resource *and* the *FinishedProduct is the output* of *BSP- Produce_Intermediate_Stage.*
- *BSP-Produce* pattern can be connected with *BSP-Waste* pattern to illustrate the waste management. The *Waste* is not modelled in this pattern. As the waste is an output resource of *Produce* service, it is possible to connect *BSP-Produce* with *BSP-Waste* pattern through *Waste*.

4.4.2 Inbound Logistics

We demonstrate another sample pattern for the inbound logistics category. According to Porter, inbound logistics involve relationships with suppliers and includes all the activities required to receive, store, and disseminate inputs.

Example 2: Inventory of raw materials

Pattern Name:
BSP-Inventory_Raw_Material

Description:
There are different stages of inventory management in manufacturing company. Here we consider the inventory management of raw material. Inventory management of raw materials is a vital part of a company.

Problem:
How do we make a business service model for storing of raw materials?

Assumptions:
We assume that the storing of raw materials as an atomic activity. It uses only the minimal resources.

Pattern Structure
The BSP for inventory management for raw material is based on Conversion specification. All the constraints of conversion pattern structure are to be satisfied when deriving the pattern for storing of raw material. Based on the assumptions, we derive the pattern, by annotating the Conversion structure.

Solution:
InventoryRawMaterial is a conversion service which *uses warehouse space* and *consumes* the *raw material*. The *InventoryRawMaterial* adds value (*produce* realtionship) to the *Raw material*. Based on the design choice discussed in chapter 3, we depict only one container for the raw material. There are possible supporting services when keeping a inventory. For example controlling climatic data and providing security. We distinguish these activities as enhance services and discussed

in chapter 5. On the other hand, the inventory management includes several activities such as receiving and issuing of raw material. These sub-activities are not modeled in this stage (It is available in Appendix B). Figure 4-15 shows the graphical picture for the inventory of raw materials.

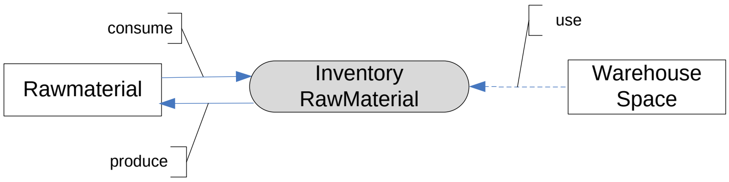

Figure 4-15: Business service pattern - Inventory of raw material

Reference:
- *BSP- Inventory_Raw_Material* pattern can be connected with *BSP-Produce* pattern using merge operator.
- *BSP-Inventory_Raw_Material* pattern can be connected with *BSP-Produce_Intermediate_Stages*
- *BSP-Inventory_Raw_Material* pattern can be connected with *BSP-Cash_Purchase* pattern.

4.4.3 Outbound Logistics

We demonstrate sample pattern for the outbound logistics category. According to Porter, outbound logistics include all the activities required to collect, store, and distribute the output.

Example 3: Inventory of Finished Product

Name:
BSP- Inventory_Finished_Product

Description:
Storing finished product in finished product store is another vital part in a production process in a company.

Problem:
How do we make a business service model for storing finished product in store?

Assumption:
We assume that the storing of raw finished product as an atomic activity. It uses only the minimal resources.

Pattern Structure
The BSP for storing of finished product basically based on Conversion specification. All the constraints of conversion pattern structure are to be satisfied when deriving the pattern for inventory management for the finished product. Based on the assumptions, we derive the pattern, by annotating the Conversion structure.

Solution:
InventoryFinishedProduct is a conversion service which *uses Warehouse Space* and *consumes* the *Finished Product*. The *InventoryFinishedProduct* adds value (*produce* relationship) to the *FinishedProduct.* There are possible supporting services when keeping an inventory. For example controlling climatic data and providing security. We distinguish these activities as enhance services and discussed in chapter 5. On the other hand, the inventory management includes several activities such as receiving and issuing of raw material. These sub-activities are not modeled in this stage. Figure 4-16 shows graphical picture for the inventory of finished product.

Reference:
- *BSP- Inventory_Finished_Product* pattern can be connected with *BSP-Produce* pattern using merge operator.
- *BSP- Inventory_Finished_Product* pattern can be connected with *BSP-Produce_Intermediate_Stages*
- *BSP- Inventory_Finished_Product* pattern can be connected with *BSP-Cash_Sale* pattern.

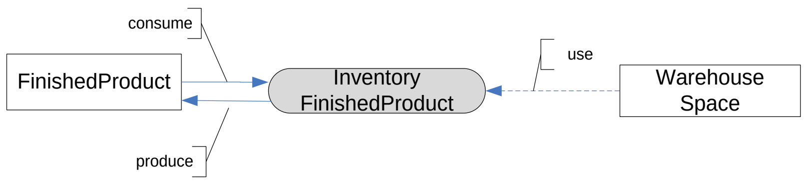

Figure 4-16: Business service pattern - Inventory of Finish product

4.4.4 Sales

This is the forth category of the primary activities. We demonstrate a sample pattern for sales in this section. Sale is a basic income generating activity of a business. We use the example of Cash sales in Hruby (2006).

Example 4: Cash Sale

Name:
BSP-Cash_Sale

Description:
Sales process is the main exchange activity which generates revenue to the company. Cash sale is the simplest form of the sales.

Problem:
How do we make a business service model for cash sale of a product?

Assumptions:
We assume that the customer pays cash at the same time of buying the product.

Pattern Structure
Cash sale follow the Exchange_Resource specification (exchange pattern structure). More commonly an external party involves in the sales. Hence, the pattern should be derived using the constraints of external exchange (goods). As we use the simplest form of sale -cash sale, annotation of Exchange_Resource - external exchange (goods) is sufficient to derive the pattern.

Solution:
Cash sale of a *Product* is an exchange service. It exchanges *Product* and *Money*. In other words, *CashSaleExchange gives Product* and *takes Money* in return. Figure 4-17 shows graphical picture for the cash sale of a product.

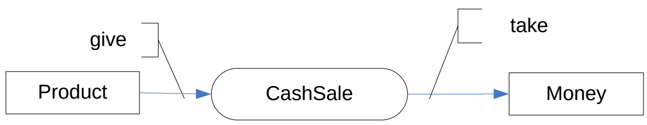

Figure 4-17: Business service pattern - Cash Sale

Reference:
- *BSP- Cash_Sale* pattern can be connected with *BSP-Product_Delivery* pattern.
- *BSP-Cash_Sale* pattern can be connected with *BSP- Inventory_Finished_Product* pattern.

4.4.5 Services (After sales services)

This includes all the activities required to keep the product or service working effectively for the buyer after it is sold and delivered. We select product repairing activity to demonstrate the after sales service.

Example 5: Product Repairing

Name:
BSP-Product_Repair

Description:
Repairing is one of the activities of after sales service provided by the company to its customers. The reaping takes place according to the terms and conditions of the sales contract.

Problem:
How do we make a business service model for repairing a sold product?

Assumptions:
We assume that repairing as an atomic activity and it uses minimal recourses.

Pattern Structure
Product Repairing follows the Conversion specification. According to assumptions, the intermediate steps are not modeled in this case. To derive the pattern, annotation of Conversion structure is sufficient.

Solution:
ProductRepair is a conversion service. It converts sold *product* which requires a repairing, into a repaired *product*. The relationship depicted as *produce* between *ProductRepair* and *Product*. The *ProductRepair* service *consumes* the sold *Product* and *Part,* and *uses Tool* to repair the product. Figure: 4.18 shows graphical picture for the product repair.

Reference:
- *BSP-Product_Repair* pattern can be connected with *BSP-Inventory_Raw_Material* pattern.
- *BSP-Product_Repair* pattern can be connected with *BSP-Product_Delivery* pattern.
- *BSP-Product_Repair* pattern can be connected with *BSP-Product_Return pattern*

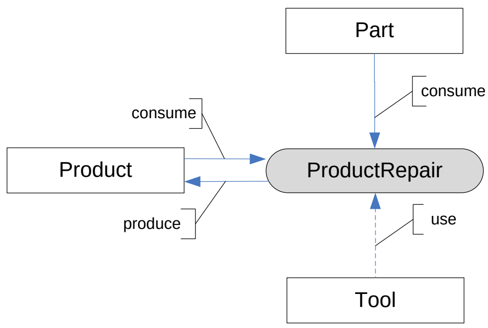

Figure 4-18: Business service pattern - Product repair

4.4.6 Procurement

As we described in section 4.4, the procurement is distinguished as a core activity of a company. Hence, we model it under the primary activities. The procurement refers to all kind of input purchasing including raw materials. We select purchasing of raw material to demonstrate the procurement with its simplest form i.e. cash purchase.

Example 6: Cash Purchase

Name:
BSP-Cash_Purchase

Description:
Raw material is the basic input of the production process. It can be purchased in purchasing department or any other department which involves in the production process. Purchasing is an exchange activity. Cash purchase is the simplest form of the purchasing.

Problem:
How do we develop a business service model for cash purchasing?

Assumptions:
We assume that focal agent pays cash the same time of purchasing the raw material. The next assumption, we consider purchasing as an atomic activity and it uses minimal recourses.

Pattern Structure
Purchasing raw material follows the Exchange_Resource (exchange pattern structure for resource) specification. To derive the pattern, annotation of Exchange_Resource specification is sufficient.

Solution:
Cash Purchase of raw material is an exchange service. It exchanges *Money* and *Raw Material*. In other words, *CashPurchaseExchange gives Money* and *takes Raw Material* in return. Figure 4-19 shows graphical picture for the cash purchase of raw material.

Reference:
- *BSP-Cash_Purchase* pattern can be connected with BSP-*Inventory_Raw_Material* pattern.
- *BSP-Cash_Purchase* pattern can be connected with *BSP-Produce* pattern.

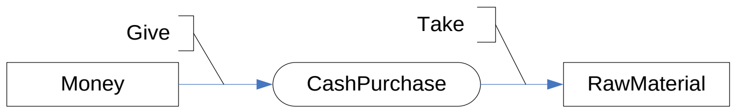

Figure 4-19: Business service pattern – Cash purchase

4.5 Generic Business Service Patterns for Support Activities

Porter defined four main areas of support activities: procurement, technology development (including R&D), human resource management, and firm infrastructure (systems for planning, finance, quality, information management etc.). As we described in the section 4.4, most of the support activities are aligned with enhance services except procurement. Porter's value chain analysis, gives general classification of firm's value activities. It doesn't provide specific examples for activities of each category. Therefore, in this section we demonstrate high level business service patterns for each category of supporting activity. However the detailed analysis of enhance services is discussed in chapter 5. In addition to the support activities mentioned above, we consider the marketing as support activity as it adds value to the product or service. In this section we demonstrate one business service pattern for each of the following activities. 1. Human resource management, 2. Technology development (including R&D), 3. Firm infrastructure, 4. Marketing. We follow the same format of pattern documentation used in primary activities.

4.5.1 Human Resource Management

According to Porter, the human resource management is a support activity. It covers all activities involved in recruiting, hiring, training, developing, compensating and (if necessary) dismissing or laying off personnel. Human resources element involves in many primary and support activities of the company. Human resource service provisioning is vital input or output of company's value creation process. Hence it is essential to pay attention to model human resource as it is a special kind of resource. The detailed investigation of human resource service provisioning is available in chapter 5. In this section we demonstrate generic pattern for the human resource provisioning.

Example 7: Employee Service Provisioning in Production

Name:
BSP-Employee_Service_Provisioning

Description:
The HR factor is available in a company by an exchange process. (according to Hruby (2006): labor acquisition event in REA). The company has to pay salary to get the service of employees. An employee is an external agent who has a contract with the organization, and provides a "service provisioning" that could be further qualified with a job description, e.g., mechanic service provisioning. Employee service enhances the product or service in the company.

Problem:
How do we derive a business service model for employee service provisioning?

Assumption:
We model employee service without qualifying the job description. (more specific patterns are described in chapter 5.)

Pattern Structure
Employee service provisioning follows the combination of Enhance pattern structure, Exchange_Service (exchange pattern structure for service) and Conversion specifications. The Enhance specification is used to model how employee service provisioning enhances the production. The Exchange_Service specification is used to model the acquisition of employee service. And the third specification (Conversion) is used to model the affected service (produce) by the employee service provisioning.

Solution:
Employee service exchange is an exchange service which gives money to take Employee Service Provisioning. The Employee service provisioning enhances the Produce service. It uses the Tacit Knowledge of employee and produces Experience as a result of engaging the production. The production activity follows the same details of BSP-Produce pattern. Figure 4-20 shows graphical picture for the Employee Service Provisioning.

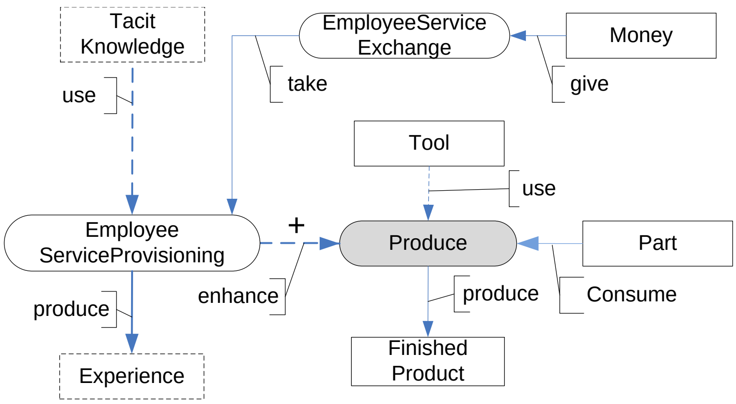

Figure 4-20: Business service pattern- Employee resource provisioning

References:
- *BSP- Employee_Service_Provisioning* pattern can be connected with any of services. (exchange, conversion , core- sub, coordination or enhance) . The job description of the employee can be qualified according to the situation.

4.5.2 Technology Development

Following Porter, the technology development affects every value activity to increase the efficiency and effectiveness. Technology development takes place in many forms. For example: office automation, research and development activities and communication technology. To demonstrate the business service model, we select *new product design* which is an R & D activity, affect to the production process. As the *new product design* improves the existing product features, we distinguish *new product design* enhances the produce service.

Example 8: New Product Design

Name:
BSP-New_Product _Design

Description:
New product design is an R & D activity in a manufacturing company. As new product design improves the existing product features, produce activity is enhanced by the new product design. The aim of most of R & D activities is improve the core activities of the company.

Problem:
How do we derive a business service model for new product design?

Assumption:
We consider the *new product design* is an atomic activity

Pattern Structure
New product design service follows the Enhance pattern structure and enhances the *Produce* service. The Produce service follows the Conversion specification.

Solution:
New product design service is an enhance service which *use DrawingTool* and *produce* the *NewDesign*. The *New product design* service *enhances the Produce service.* The production acidity follows the same details of BSP-Produce pattern. Figure 4-21 shows graphical picture for the new product design.

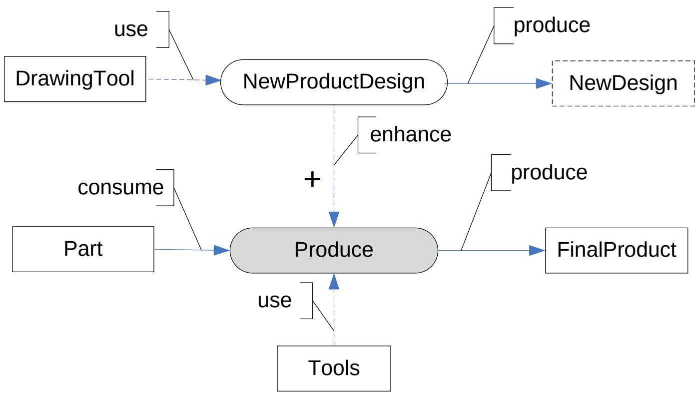

Figure 4-21: Business service pattern – Product design

References:

- *BSP- New_Product _Design* pattern can be connected with *BSP-Employee_Service_Provisioning*.

4.5.3 Firm Infrastructure

Firm Infrastructure consists of various activities including planning, legal management, finance, quality management, information management etc. Firm infrastructure supports the entire value chain in the firm. The goal of the firm Infrastructure is to increase the efficiency and effectiveness of the primary activities of a firm. As we identified support activities as enhance services, the firm infrastructure is also modeled as enhance service. On the other hand all the firm infrastructure activities represent the management practices. Hence, we further qualify the firm infrastructure as a management service. Three kinds of sub-services are identified for the management services namely monitoring, evaluating and enforcement. (detailed discussion of the management services is available in chapter 5). In this example, we demonstrate one activity which relates with quality management of a product. The quality management of the business as a whole or how it is implemented is not the goal of the following pattern. But the pattern is used to indicate how the quality management as a single activity, can be modeled as a service. Further, the BSP of product quality management shows how support service activity affects to a core service. It is possible to decompose the product quality management service into sub-services like inspection, recovery management etc.

Example 9: Product Quality Management

Name:
BSP-Product_Quality_Mgt

Description:
For any kind of business, quality management is a vital activity to be in competitive edge. Quality management includes four main activities, planning, controlling, assuring and improving. Quality management adds value to the product by increasing its quality, reducing the waste and rework

Problem:
How do we make a business service model for quality management of producing a product?

Assumption:
We assume product quality management as a single composite activity.

Pattern Structure
Product Quality Management service follows the Enhance pattern structure specifications. The service which is affected by the quality management follows the

BSP-Produce pattern which is based on Conversion specification. As there are no sub-services, the pattern structure annotation is adequate to derive the new pattern.

Solution:
Product Quality management is an enhance service which *enhances* the *Produce* service. *Product Quality management* uses intentional resource for example, Quality measurement parameters - *QulaityParameters* and it is possible to produce another intentional resource called *Recovery Action List*. Figure 4-22 shows the graphical picture.

Reference:
- The *BSP- Product_Quality_Mgt.* pattern can be connected with any of services (exchange, conversion, core-sub, coordination or enhance) with *enhance* relationship.

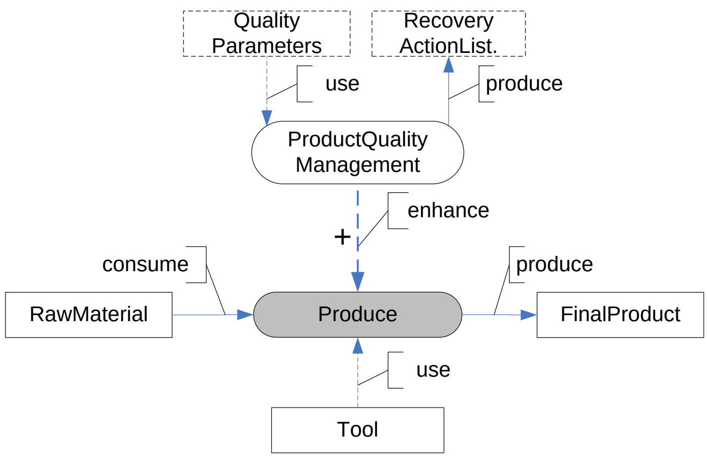

Figure 4-22: Business service pattern – Product quality management

4.5.4 Marketing

As we mentioned in section 4.4, Porter defines marketing under primary activity. But our view on marketing deviates from this definition. As marketing improves the sales process by persuading customers, we distinguish marketing as a supporting activity. According to the American Marketing Association (AMA, 2012) board of directors, marketing is the activity, set of institutions, and processes for creating, communicating, delivering, and exchanging offerings that have value for customers, clients, partners, and society at large. Among several activities of marketing, advertising and sales promotion are two examples. Advertising mainly focus on publicity of the product or service, where as the sales promotion consists of a diverse collection of incentive tools such as coupons, prizes, cash refund, warranties, demonstrations, buying allowances, free goods and bonuses. According to the definition of AMA (2012) sales promotions are media & non media marketing pressure applied for a predetermined, limited period of time in order to stimulate trial & impulse purchases, increase consumer demand or improve product quality. The sales promotion can be implemented in several ways. In this example, we demonstrate one activity – *Warranty* as a sales promotion activity to represent the marketing management. The intention of the pattern is not to model the whole marketing management process or the sales promotion strategies of the business. But the pattern demonstrates how a single marketing management activity is modeled as a service. The warranty service enhances the core service of a business. The BSP for the warranty service of a product is described below. The BSP for advertising is discussed in chapter 5.

Example 10: Product Warranty Service

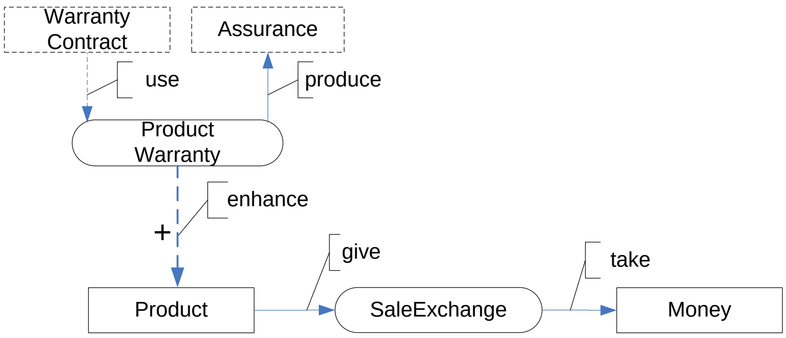

Figure 4-23: Business service pattern – Sales promotion

Name:
BSP-Product_Warranty

Description:
The warranty is assurance from one party to another party. The warranty of a product is a legal assurance for the product which provides by a seller or the manufacturer to the customer. The customer who buys the product has a guaranty to the product for a specified time under certain conditions which are described in the warranty contract. Hence, it stimulates the purchasing power of customer.

Assumption:
We assume the product warranty as a single activity.

Problem:
How do we make a business service model for product warranty?

Pattern Structure
The product warranty follows the Enhance pattern structure specification. The affected resource is the product.

Solution:
The Product warranty is an enhance service which enhances the *Product*. The *Product warranty* service uses *Warranty Contract* which is defined by the company and produces *Assurance*. The *Warranty Contract and* the *Assurance are* intentional resources. Figure 4-23 shows the graphical picture for BSP-Product_Warranty.

4.6 Business Service Patterns for Service Industry

Porter's notion of the value chain is useful for analyzing manufacturing industries. The similar approach, perhaps as a generalization of Porter's, does not facilitate the analysis and categorization of the wide variety of service industries (Nooteboom, 2007). The review of generic Porter's value chain done by Gabriel (Gabriel, 2006), shows several differences in manufacturing and service industries as listed next few lines. In the service industry, the real operation, inbound or outbound logistics are not visible in the same way in manufacturing industry. In the manufacturing industry, the operations can take place in isolation of the customer. In the service industry, production and usage of the service occur simultaneously. The infrastructure required in the service industry might also be different from that of the service industry. The supporting activities might be similar in most cases, yet the way of implementing managing the supporting activities might be different. Stabell and Fjeldstad (1998) also state that the Porter's value-chain model does not sufficiently capture the value creation logic of service industries. They suggest that more than one model is needed to understand the workings of the different types of business and propose three distinct value configurations namely value chain, value shops and value chain. Therefore, we propose a generic business service pattern and several specialisations to specific sectors for service industry.

Figure 4-24 shows an enterprise level generic BSP for a service business. Selling of a *Service* is the main activity of generating income. It is modelled as *SaleOfServiceExchange* which gives a *service* (we call it tradable service hereafter) in return of *money*. The service may have sub-services and those are not modelled in the figure. The tradable service always affect to some resource / service of the customer's side. It is depicted as *PhysicalResource1* in the diagram. There are several ways to generate the tradable service. The service can be outsourced or generated within the company or both. Example for outsourcing is a vehicle renting company who hires vehicles from third party and rent them to customers. Outsourcing is depicted in the figure *OutsourcingServiceProcureExchange* which takes the *OutsourcedService* and gives *money* in return. Example for the service generated within the company is hair cutting service. The haircutting service uses / consumes physical resource like scissor / shampoo and gets the service of hair cutter. The hair cutter's service is an enhance service which is a paid service. It is possible to have coordination services in both cases. We demonstrate business service models for several categories of service industry in next sections. We follow some of the categorization details service industry provided by (Nooteboom, 2007). As the service patterns represent enterprise level, we use the prefix ESP (Enterprise service pattern),

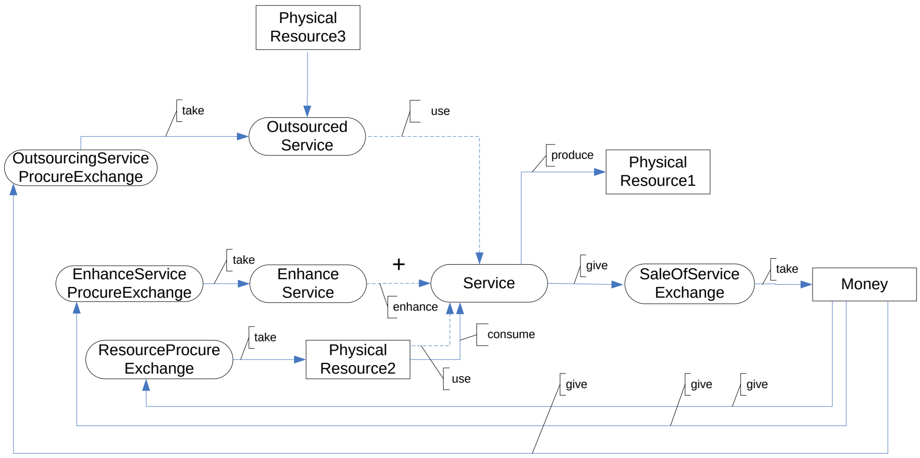

Figure 4-24: Generic business service pattern for service industry

4.6.1 Utility services

Utilities entail the supply of gas, water, electricity, telephone service, internet access and cable TV (Nooteboom, 2007). In this case, we consider only the distribution of utility service. The distribution of utility service includes flow of materials using pipes, cables or the network. The production of these services is not modeled here.

Name:
ESP-Utility_Service

Description:
There are different types of utility services. Electricity and gas are good examples. The distribution of the utility service requires a medium to distribute (e.g., cables), equipments and the technical staff. The customer is buying the utility service with a certain fee.

Problem:
How do we make an enterprise service model for a utility service?

Pattern Structure
Utility service follows the Exchange_Service (exchange of service) and Exchange_Resource specifications. The first specification is used to model the sale of utility service and the second one is used to model the purchasing of utility resource to the distributing company. The distribution follows the Conversion specification. The Enhance specification is used to model the technical staff service provisioning.

Solution:
The Figure 4-25 shows the enterprise service model for distribution of a utility service. *Sale* is the main activity which generates *money* to the company by distributing the utility resource. The model follows Exchange_Service specification to build the *Sale* service. Following Conversion specification, the *UtilityServiceDistribute* consumes the *UtilityResource* which is outsourced and adds value to the *UtilityResource* (produce relationship). The distribution of the utility service may need technical staff and technical equipment. Therefore, the model has *enhance* relationship with *TechnicalStaffServiceProvisioning* and *use* relationship with *equipment.* The technical staffs have to be acquired to company by an exchange service. Hence, the model has *TechnicalStaffServiceExchange* service to represent the acquiring of the technical staff.

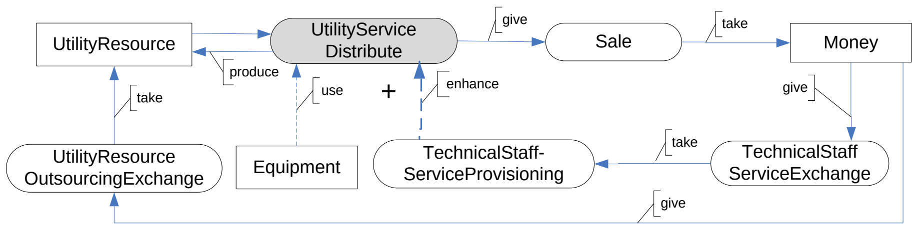

Figure 4-25: Business service pattern – Utility service

4.6.2 Transportation Service

The movement of people or objects from one location to another is the objective of the transport service. It may have soft goals such as comfort of travel, safety travel, reliability of travel schedules and speediness etc. The transport service may have coordination services for example reservation.

Name:
ESP-Transport_Service

Description:
The passenger transport is a service which changes the location of the passenger. There are several number of transport mediums. For example plane, ship, bus or car. The realization of the service in each of the example is different.

Problem:
How do we make an enterprise service model for the passenger transport service?

Pattern Structure
Transport service follows the Exchange_Service (exchange of service) to model the sale of transport service. As the transport itself a conversion service, we follow the Conversion specification. The Enhance specification is used to model the transporting staff service provisioning. The Exchange_Resource specification is used to model the purchasing of vehicle.

Solution:
The Figure 4-26 shows the passenger transport service. The *transport* service is provided by the company to the passenger in return of *money*. We follow the Exchange_Service to model the *TransportExchange*. The *Transport* service adds value to the *Passenger* by changing his location. Hence, the *Passenger* is affected by the *Transport* service. As the passenger is a passive role, we model him as a physical resource. The *Transport* itself uses the *vehicle* and the driver's services. The money has to be spent to buy vehicles and acquire driver's service. These two concepts are modeled in the figure as *VehicleProcureExchange* and *DrivingServiceExchange*.

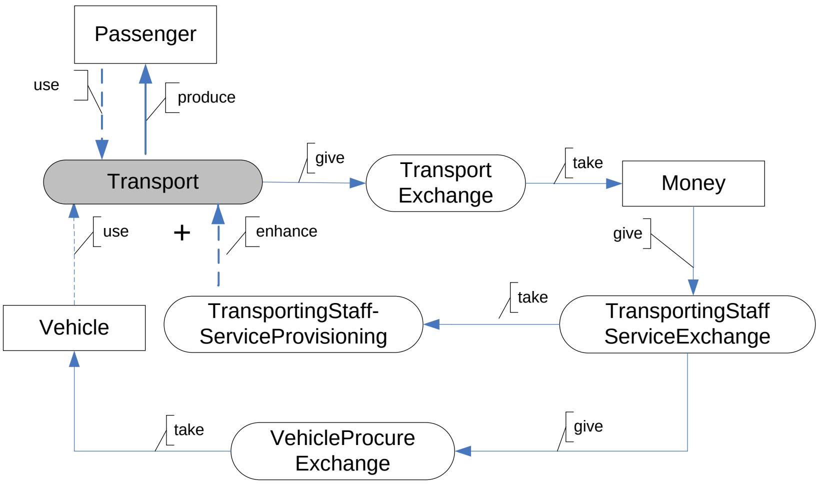

Figure 4-26: Business service pattern – Transport service

4.6.3 Personal care services

The personal services include health and other care (hairdressing, cosmetics, etc.), recreation, professional knowledge services (e.g., consultants, accountants, etc.) and education. These all affect either the body or the mind of people, or both, and yield utilities of physical or mental wellbeing or capacity (Nooteboom, 2007). We demonstrate a healthcare service under personal care.

Name:
ESP-HealthCare_Service

Description:
The healthcare is one category of personal care. There are different types of healthcare. We demonstrate generic enterprise service pattern for the healthcare. The patient is treated by the healthcare service using medical equipments and medical drugs. Different levels of medical staff engage in the healthcare service such as physician, nurse, chemist and physiotherapist etc.

Problem:
How do we make an enterprise service model for the healthcare service?

Pattern Structure
The healthcare service follows the Exchange_Service (exchange of service) to model the sale of healthcare service. As the healthcare itself a conversion service, we follow the Conversion specification. The Enhance specification is used to model the medical staff service provisioning. The Exchange_Resource specification is used to model the purchasing of drugs and equipment. .

Solution:
The Figure 4-27 shows the BSP for the healthcare service provider. The *Healthcare* service is the main service provided by the healthcare provider in return of *money*. The *Healthcare* service adds value to the *Patient*. Hence, the *patient* is affected by the *Healthcare* service and he is a passive role. As the *Patient* is a passive role, we model him as a physical resource. The Healthcare service is enhanced by the *MedicalStaffServiceProvisioning.* The medical staffs exist in the company by acquiring. It is depicted in the model as *MedicalStaffServiceExchange*. The Healthcare service uses *MedicalInstruments* and consumes *Drugs*. The possible sub-services to the *Healthcare* are medical examination, physiotherapy,

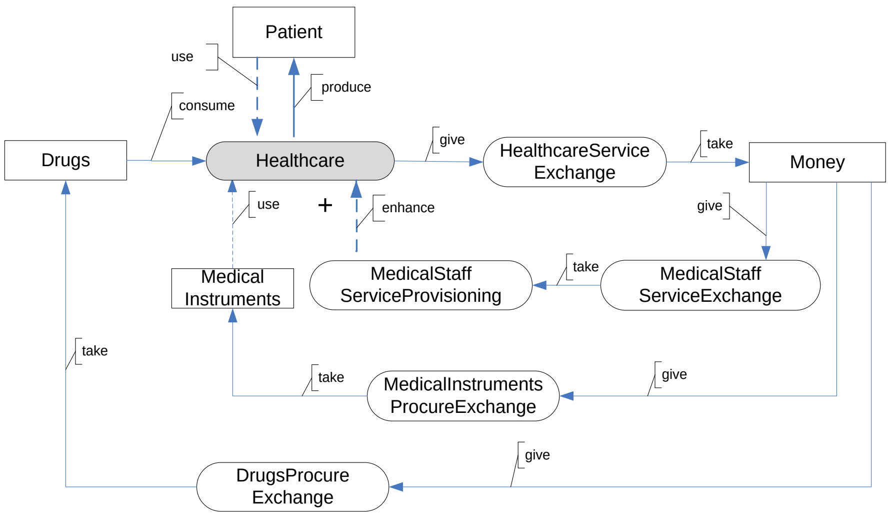

Figure 4-27: **Business service pattern - Healthcare service**

4.6.4 Insurance services

Insurance is a service which deals with risk management. The insurance companies are selling the insurance and charge certain amount of money for their service. The person or object can be insured with an agreed policy. There are several categories of insurance for instance life insurance, healthcare insurance, insurance for fire and natural disaster and vehicle insurance etc. The amount to be charged for a certain value of the insurance coverage is called the premium. In this section we model healthcare insurance.

Name:
ESP-Health_Insuarance_Service

Description:
The health insurance is a common insurance which covers the healthcare activities of the insured person. There are different types of healthcare insurance. We demonstrate generic enterprise service pattern for a basic healthcare insurance. The client is insured by the insurance service. The insurance company gets a premium for the service. The insurance company has to pay for the agreed healthcare activities of the client. The staffs of the insurance company help to provide the service.

Problem:
How do we make an enterprise service model for the health insurance service?

Pattern Structure
The healthcare service follows the Exchange_Service (exchange of service) to model the sale of insurance service. As the health insurance itself a conversion service, we follow the Conversion specification. The Enhance specification is used to model the insurance staff service provisioning. The Exchange_Resource specification is used to model the purchasing of drugs and equipment. The insurance company pays money to the healthcare services of client. We model it using Exchange_Service specification.

Solution:
Figure 4-28 shows the business service pattern for the healthcare insurance service. The *Insure* is the service provided by the *InsuranceServiceExchange* service and the company gets a certain amount of *Money* which is called the premium (follow the Exchange_Service specification). The *Insure* service uses *Documents* as a physical resource. The service of the insurance staff adds value to the *Insure* service. The staff has to be acquired to the company with a fee. The *InsuranceStaffServiceExchange* shows the acquiring of the staff. The *Insure* service has a sub-service called *HealthcareServiceExchange*. The *HealthcareServiceExchange* service takes *Money* and gives it to a service which is in client's side. Here it is the *Healthcare service* which adds value to client. As the *Insure* service adds value to the client by providing health insurance. There is a *produce* relationship between the *Insure* service and the *Client*.

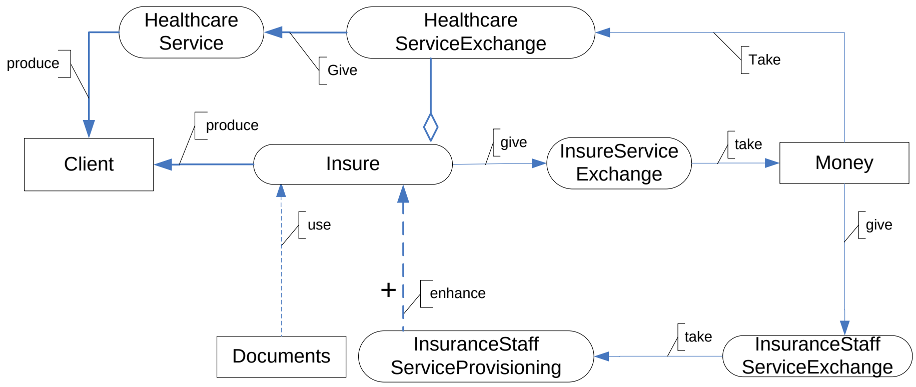

Figure 4-28: Business service pattern – Insurance service.

4.7 Pattern Composition with Business Service Pattern Operators

The enterprise model is a composition of BSPs that always starts from a generic enterprise model (for trading company, for manufacturing etc). Some of the generic enterprise models are described in Appendix C. In this section we define business service pattern operators (BSPOs). The purpose of using BSPOs is combining two or more BSPs together. The pattern combining approach using BSPOs offers several opportunities. Primarily, it allows to create new patterns using existing ones. Then, it provides an opportunity to compose the enterprise business service model using BSPs. Each time when the BSPOs are used, the enterprise model is expanded.

We provide an operator called *Merge,* to combine one or more business service patterns. The term "pattern" is used for both the basic patterns given in the library and the compositions of patterns. Again, we follow the pattern composition mechanism in Botton, et al. (2010). According to the formal approach of pattern composition in Bottoni et al. (2010), it is required to define the models as graphs. As all the pattern structures are based on the definition of *variable pattern* (c.f. 4.1), the pattern structures and the patterns can be viewed as graphs. Each pattern / graphical structures of pattern specifications can be expressed with nodes and edges. The graphical view of the patterns in ConceptBase, shows the graph structure of the pattern. Hence, we can apply the mechanism used in Bottoni et al. (2010), for the pattern composition. The formal description of *Merge* operator is given below

4.7.1 Merge Operator

Description:

The merge operator enables to combine two patterns and compose a new pattern. We follow the definition b-4 in Bottoni et al. (2010) in which the composition is described using the technique called "pushout" by gluing the two objects along a common object or group of objects.

Parameters :

BSP1, *BSP2*: business service pattern 1 and 2 respectively;
K: common object to *BSP1* and *BSP2*;
m1,m2 : Morphisms *m1, m2* to *BSP1* and *BSP2*:
$BSP1 \xleftarrow{m1} K \xrightarrow{m2} BSP2$ respectively.

Constraints:

1. Take the union of the constraints related to the common object (Constraints of K in BSP1) ∪ (Constraints of K in BSP2)
2. All the other constraints remain unchanged.

Usage :

1. Using Merge Operator in Pattern Composition
2. Using Merge Operator in Pattern Decomposition

Candidates for common object:

The possible candidates for the common object can be a *physical resource*, an *intentional resource,* a *conversion / exchange service,* a *coordination service* or an *enhance service*. It is possible to have a set of common object.

4.7.1.1 Using Merge Operator in Pattern Composition

The simplest way of using *Merge* operator is used in direct pattern composition. A new pattern or model can be composed by merging two or more patterns. In this case, the merge is used when directly gluing the two or more patterns along a common object or group of objects. The following example (Figure 4-29) illustrates the merge operation. We demonstrate the composition of delivery pattern (BSP_Delivery) and delivery outsourcing pattern. Assume that the company needs to outsource the delivery service when self delivery is not possible due to lack of resources. This situation creates a need to incorporate delivery outsourcing pattern to the existing model. We use the *Merge* operator to combine these two.

Pattern (a) in Figure 4-29 represents the product delivery service. *Delivery* is a conversion service *used* in the *Sale* service. *Delivery* service *uses Truck* and *ForkLift* (which are physical resources), and it adds value to the *Product* by changing its location.

Pattern (b) in Figure 4-29 shows the delivery outsourcing. It is an exchange service (*DeliveryOutsourcingExchange*), which *takes* the *Delivery* service and *gives Money*. *Delivery* service in the outsourcing pattern *uses Truck*. Merging product delivery with delivery outsourcing is done through the common group of objects "*Delivery*" plus "*Truck*". Note that the relationship of *Delivery* and *Truck* is also same in pattern (a) and (b). Figure 4-29(c) shows the merged pattern for product delivery outsourcing.

As we merged two patterns, it is required to re-build the constraints of the new model as a union of constraints which relate to the common object/ group of objects (in this example *Delivery* service and the *Truck*). We use ConceptBase query classes to define constraints in pattern structure specification. As the ConceptBase query classes do not support for merging, the implementation is not complete. One of the possible solutions is using ConceptBase Active rules to merge two patterns. This part of implementation is in future agenda. However, we explain the merging constraints in English as follows.

Constraint of common group of objects in product delivery pattern Figure 4-29.
((*DeliveryService* **use** OutflowResource) **and**
(*DeliveryService* **produce** InflowResource) **and**
(InflowResource ≠ OutflowResource)) → Statement (1)
Outflow resource = Truck, Fork Lift
Inflow resource = Product

Constraint of common group of objects in delivery outsourcing pattern (cf. Exchange_Outsource specification):
((OutflowResource **control** ExternalAgent) **and**
(InflowResource **control** FocalAgent) **and**
(InflowResource ≠ OutflowResource)) → Statement (2)
Outflow resource = Truck,
Inflow resource = Product

These constraints are not inconsistent. The union can be written as:
{((*DeliveryService* **use** *Truck*) **and**(*Truck* **control** ExternalAgent)) **and**
((*DeliveryService* **produce** *Product*) **and** (*Product* **control** FocalAgent))}→
Statement (3)

The statements (1, 2 and 3) are explained below with details.

Statement (1):
The *Delivery* service *uses* at least one Outflow resources (*Truck and Folk Lift*) and *produces* at least one Inflow resource (*Product).*

Statement (2):
According to the Outsource specification, at least one resource outflow is controlled by an external agent and resource at least one inflow resource is controlled by Focal Agent. With regards to outsourced *Delivery* service, the statement can be expressed as follows. The *Truck* is owned by the *external agent* and the *Product* is owned by the focal agent.

Statement (3):
With regards to merged pattern (Product delivery outsourcing service), the statement can be expressed as follows.

"The *Delivery* service *uses Truck* and the *Truck* is owned by the external agent and also the *Delivery* service *produces Product* and the *Product* is owned by the focal agent."

Therefore, once a service is outsourced, the outsourced service affects at least one resource of the focal agent. At the same time the outsourced service always uses at least one resource which belongs to the external agent.

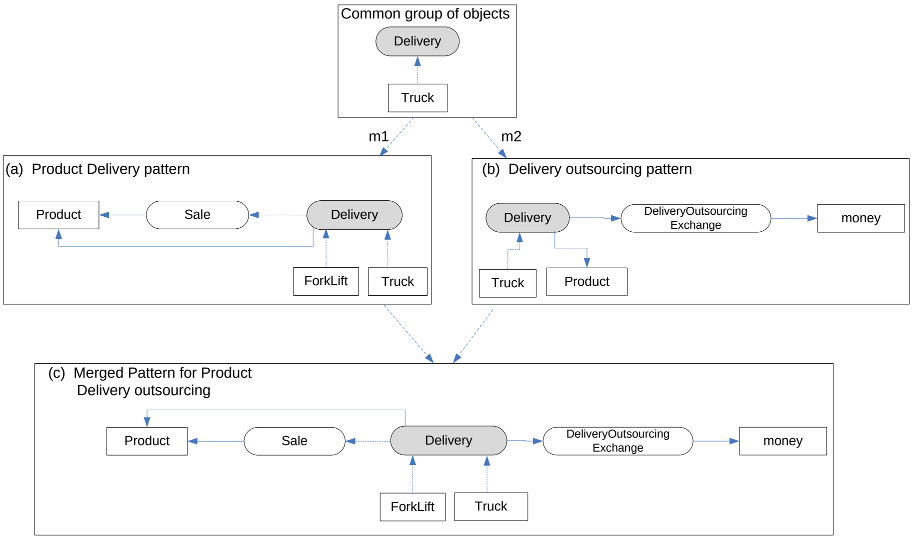

Figure 4-29: *Merge* operator in pattern composition

4.7.1.2 Using Merge Operator in Pattern Decomposition

The *Merge* operator is used not only in pattern composition but also in decomposition. Simple meaning of merge is joining together. Decomposition means braking down into several pieces. Even the meaning of merge and decomposition gives opposite directions, the merge operator is used in the pattern decomposition. In this section, we explain the pattern decomposition using *Merge* operator.

Decomposition is necessary when a service has to be realized by multiple activities. The objective of decomposition is visualizing the underlying services of a composite service. The process of decomposition is described with three steps as follows. In the first two steps, we use pattern operations (expansion and annotation) which were described in section 4.3. The merge operator is used in the third step.

Step 1: Pattern structure expansion (if there are more than 2 sub-services)

First step deals with Sub-Service pattern structure only. Based on the number of sub-services, the pattern structure has to be expanded.

Step 2: Derive the domain specific model

The second step is deriving the domain specific model using pattern structure annotation. Pattern annotation is used the domain specific pattern using specialize vocabulary in a specific domain. The structure derived in the previous step is used as the source graph..

Step 3: Merge the composite service pattern with sub-service pattern

The third step is merging the domain specific sub-service pattern with the composite service pattern using *Merge* operator.

We use an example of bike producing to demonstrate the *Merge* operator in decomposition. Assume that bike producing has assembling, painting and inspecting sub-activities. According to the first step, the sub-service pattern structure has to be expanded as the number of sub-services is more than 2. Then we move to the step 2. By adopting a vocabulary pattern which consists of the names of specific domain-bike producing, we derive domain specific sub-service pattern for the bike producing (Figure 4-30(b)). *BikeProduce* has three sub-services namely, *Assemble*, *Paint* and *Inspect.* These sub-services are connected to the *BikeProduce* service with *part of* relationship. Following the constraints of Sub-Service specification, *BikeProduce* and the sub-service- *Assemble* use *Frame* and *Wheel* as inputs. The final output which is *FinishedBike* is produced by the *Inspect* sub-service. It has the same relationship with the composite service as well. The intermediate resources are produced and consumed by the sub-services as depicted in the Figure 4-30(b).

In third step, we are merging composite service pattern which is in Figure 4-30(a) with the sub-service pattern of bike producing (Figure 4-30(b)). *BikeProduce* service, *Frame*, *Wheel* and the *Bike* are the common group of object for both. The Figure 4-30(c) shows the decomposed service pattern for bike producing.

As we merged two patterns, it is required to re-build the constraints of the new model as a union of constraints which relate to the common group of objects

Constraint of common group of objects in product delivery pattern Figure 4-29.

((*BikeProduce* service **consume** OutflowResource) **and**
(*BikeProduce* service **produce** InflowResource) **and**
(InflowResource ≠ OutflowResource)) →Statement (1)
Outflow resources = Frame, wheel
Inflow resource = Bike

Constraints for sub-service pattern:

- At least two sub-services are to be defined.
  ((*BikeProduce* service **part-of** *Assemble* service) **and**
  (*BikeProduce service* **part-of** *Paint* service) **and**
  (*BikeProduce service* **part-of** *Inspect* service)) →Statement (2)

- Inflow resources of the composite service must be consumed by a sub-service, outflow resources of the composite service must be produced by a sub-service
  ((*Assemble* service **consume** OutflowResource) **and**
  (*Inspect* service **produce** InflowResource)) →Statement (3)
  Outflow resources = Frame, wheel
  Inflow resource = Bike

The union of above three constraints can be written as:
[(*BikeProduce* service **part-of** *Assemble* service) **and**
(*BikeProduce service* **part-of** *Paint* service) **and**
(*BikeProduce service* **part-of** *Inspect* service)] **and**
[(BikeProduce service **consume** Frame, Wheel) and (*Assemble* service **consume** Frame, Wheel) **and**
(*BikeProduce* service **produce** Bike) **and** (*Inspect* service **produce** Bike)]

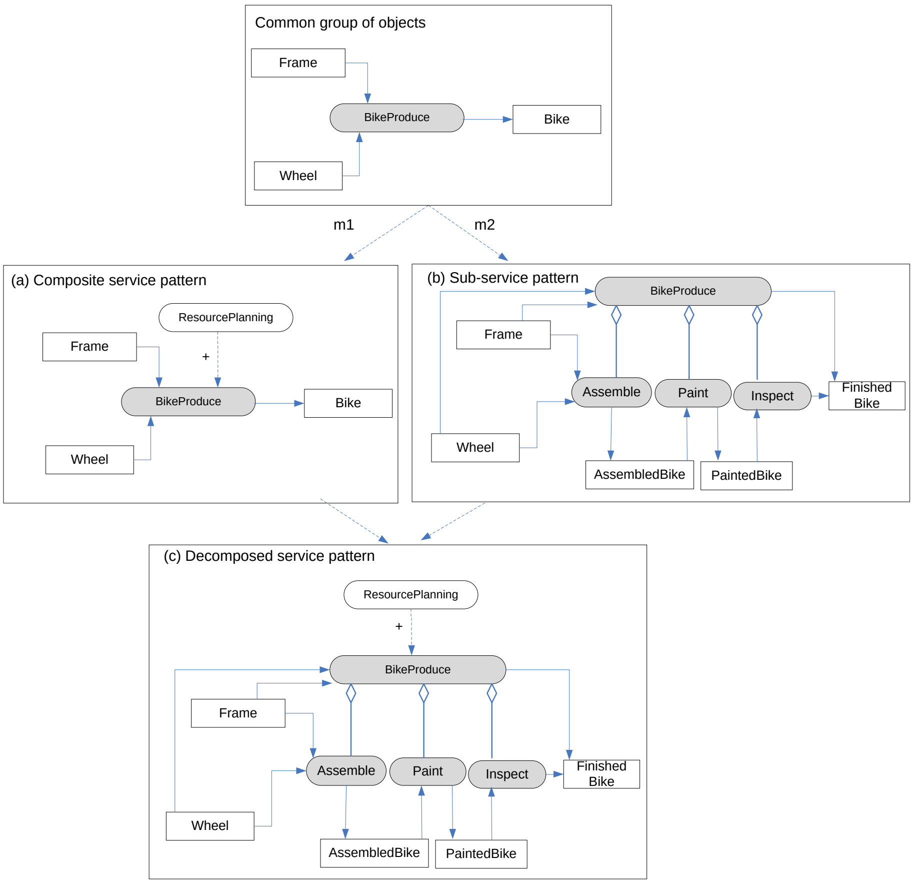

Figure 4-30: *Merge* operator in decomposition

4.8 Design Steps for Enterprise Information Systems

Given the service patterns and pattern operators, a service design steps can be developed. The following activities are intended to describe the design steps to achieve such model. All the steps after the first step are not strictly ordered. The design process starts from the first step and there after follows incremental approach where more than one iteration has to be performed

Step 1: Ø → enterprise model
Step 2: decomposition: $\{s_0\} \rightarrow \{s_0, s_1, \ldots s_n\}$, part-of($s_i$,s_0)
Step 3: coordination: $\{s_{1,\ldots,} s_n\} \rightarrow \{ s_0, s_{1,\ldots,} s_n, c_i,\}$, part-of($c_{i,}$, s_0), coordinate($c_{i,}$, s_0)
Step 4: extension: $\{s_{1,} s_2\} \rightarrow$ merge ($s_{1,} s_2$)
Step 5: enhance: {s} → {s,e} enhance(s,e)
Step 6: specialization
Step 7: outsource, reengineer

Step 1: The design starts top-down with the generic enterprise service model, to ensure the completeness. Generic enterprise service models for several categories are described in Appendix C. For example manufacturing, trading, service sectors.

Step 2: In the second step, identify specific services by decomposing the enterprise model. As we described in the decomposition with merge operator in the previous section, decomposed service pattern with sub-service can be derived.

Step 3: Once multiple sub-services exist, they have to be coordinated. Hence third step introduces the coordination service/s.

Step 4: In the fourth step allows to extend the model. For instance, there may be a manufacturing service pattern that generates waste and a waste management pattern, then the first one can be extended by merging it with the second.

Step 5: In this step, enhance services are adding to the model, such as management services, to other services where necessary. Note that the enhance services can be decomposed, enhanced, etc. as well.

Step 6: This step allows to specialize the services /resources where ever possible. This is an important step which allows to customize the business solution. The business service model allows a room to customize further according to the special requirements using the operation of pattern annotation. For example the bike producing (which is discussed in the validation chapter) can be customized into producing of mountain bike.

Step 7: This step specifically supports model evolution. Model evolution includes outsourcing services, but also reengineering in the form of the reversal of any step 2-6. Our aim is to support meaningful model evolution steps, not arbitrary deletions and insertions.

As a summary, this chapter introduced a pattern based enterprise modeling mechanism with business services. We follow the formal approach for pattern based modeling that was proposed by Bottoni et al. (2010). The patterns are derived according to one or more specifications of five service categories. The specification of the pattern structure describes the basic requirements and the graphical structure of the pattern.

The pattern based approach for business service modeling proposed in this chapter, is flexible enough to adopt the changes. The pattern operations allow to expand the model by merging. The customization of patterns to a specific domain is defined using pattern structure annotation. More importantly, the global sourcing of resources is one of the key success factors of a business. The *Outsourcing* pattern supports the idea of global resource sourcing

Finally we propose design steps to compose patterns into an enterprise model. We validate the proposed pattern based service modeling approach with two case studies (Chapter 6).

4.9 Service Integration

Smooth integration of services is one of the key benefits of service- oriented enterprises. The authors of Allgaier et al. (2010) say that SOA environment creates an opportunity to organizations to interact dynamically as service consumers and service providers making use of a service marketplace to design, offer and consume services. But the key questions are "Does the selected service address the real business need?" and "How to select the right service?" Separating the business choices and technical choices is necessary in order to structure the solution to these questions. The role of business service pattern is not limited to design model. It can also be used in the discovery of the services in a service marketplace or service library. Based on business service patterns, we propose a business service integration mechanism with software services. First, we discuss stat of the art of the pattern based service integration approaches (section 4.9.1). Then section 4.9.2 describes the BSRM and process model mapping. We propose a metamodel for service integration and it is described in section 4.9.3. The last section demonstrates an example.

4.9.1 State of the Art

There exists some relevant work on pattern-based service integration. The research in Allgaier et al. (2010) presented a pattern-based modeling approach to achieve the unforeseen integration of services into extensible enterprise systems. This framework demonstrates service integration to the presentation (HCI) layer using adaptation patterns that group common patterns of model elements and their relationships. Even though this approach enables an integrator to model or design the relevant integration aspects on a higher abstraction level than implementation-level, service thinking at the business level is not addressed. The work is related to the concept of plug-in technologies that allow the development and installation of web 2.0 applications.

Based on the example of SAP's Enterprise Services, authors of Roy et al. (2010) describe a representational model that integrates both service and data by consolidating existing models and patterns used during the service design process. On top of this model, they created a metadata repository based on a list of ES and their respective metadata. Both representational model and metadata repository represent the basis of the iterative search of software services. Even though the business objects are one of the ingredients of the representational model, there is no clear basis of selecting business objects to their model and their focus is on the presentation layer of the ES by providing a pathway to service matchmaking.

There is also relevant research that incorporates business thinking to the service design and aims at mapping those designs to the software level. One good approach is value based service design. Zdravkovic and Ilayperuma (2010); Weigand et al. (2009); De Castro et al. (2009) present an MDA approach to design and transform services from CIM, PIM to PSM. However, the focus of all these approaches is on service design and transformation, not on service integration. On the other hand Allgaier et al. (2010) and Roy et al. (2010) have a focus on service integration. One problem is that business choices and technical choices are all dealt with together at a technical layer and have an exclusive software engineering perspective.

4.9.2 Metamodel for service integration

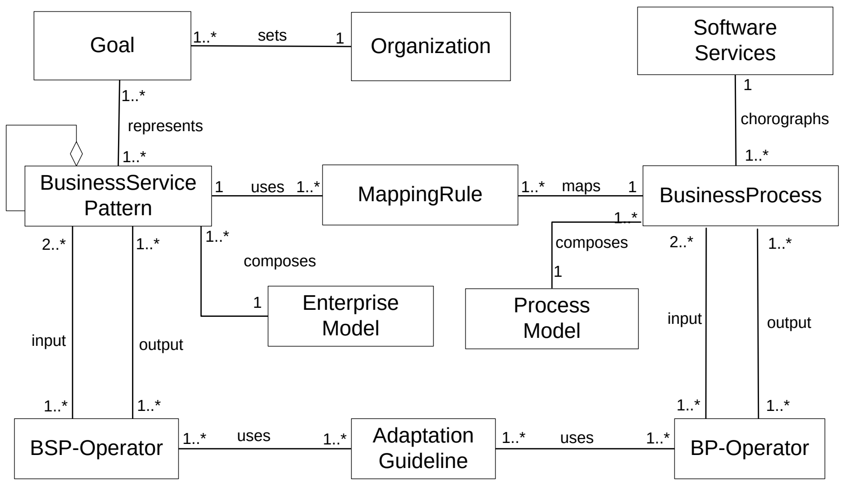

Figure 4-31: Metamodel for the pattern-based service integration

Figure 4-32 shows the metamodel for composing the enterprise model and how the business service patterns are related in the service discovery. As soft goals (extra-functional requirements) should be optimized, service discovery should address these goals. Hence these goals are represented in the BSPs: some BSP may prioritize efficiency, another customer-intimacy. Between the BPSs and the business processes a mapping exists, as discussed in chapter 3. When using the BSPs in a service marketplace setting, these mappings may also be defined manually. As explained in section 4.7.1, BSPs can be combined using algebraic operators such “merge”. Each time the enterprise model is transformed, using a BSP and BSP operator, the corresponding business processes (BPs) are integrated as well, so that when finishing the enterprise model, the designer also has an integrated BP (to be more precise: a set of possible BPs as we assume that the enterprise model still leaves room for different process implementations). The adaptation guide lines describe the homeomorphism between BSPS operators and BP operators, including conditions and their use and pragmatic guidelines to the designer.

Adaptation Guidelines:

Rule 1:

For every merge operator in the BSP, there are one or more message lines in the BP.

Conditions:

If the common object of the merge operator is a service:

- If the service exists internally, the message lines run in between swim lanes of same pool.
- If the service belongs to another party, the message lines run in between swim lanes of different pool/s.

These guidelines provide business process to derive the right parameters which is useful to find out the required library item.

4.9.2.1 Example

For example, the soft goal of customer intimacy can be increased by delivering the product to the customer at right time. It can be achieved by having an option of outsourcing the delivery service when self delivery is not possible due to lack of resources. This requirement can be implemented by merging the basic "Product Delivery" pattern (Figure 4-29(a)) with "Delivery Outsourcing" pattern (Figure 4-29 (b)). These two patterns are combined using a merge operator with the common node e.g., Delivery (composite pattern is shown in Figure 4-29 (c).

Corresponding change in the business process side can be viewed as creating a relationship by means of message line ("delivery request") from the company to the external agent. It is the operator in business process side.

Chapter 5

Enhance Services

5.1 Introduction

Every service has two main goals: satisfy the customer (who interacts with the service) and satisfy the business (that has set up and maintains the service). Apart from the cost effectiveness for both parties, several other factors play a major role to achieve the above dual goals. For the customer quality, flexibility, convenience and intimacy are few examples. Possible examples for the business are efficiency, effectiveness and security. Hence it is important to indentify these second order values with corresponding to extra-functional properties. On the other hand Porter's value chain analysis shows that the systematic linkage of primary and supporting activities is crucial factor to gain competitive advantage (Porter, 1985). According to Porter, extra-functional features are achieved through supporting activities. He specifically mentioned indirect activities (activities that make it possible to perform direct activities on continuing basis such as sales force administration, maintenance) and quality assurance activities (activities that ensure the quality of other activities such as monitoring, reviewing). Identifying these support activities as services has several advantages not only to the business users but also to the application designers. Taking necessary actions to further improvements, allocating separate budget Chesbrough (2011), defining organization constraints to these activities are some of the advantages for business users**.** From design perspective, this separation increases the modularity and the granularity for these services. Most of these supporting activities are often mixed up with the operational service logic itself, or it is handled in a separate not service-oriented system (Weigand et al., 2011). Therefore, it is important to relate these support activities to the business service model.

We identified some commonalities between the enhance services and the support activities defined by Porter. For example both have an existence-dependency. The supporting activities affect at least one primary/operational activity. Enhance services also affect at least one operational service directly or indirectly.

In this chapter we zoom into the enhance services further. We defined enhance service as a role of a service, in our service metamodel. By definition, *enhance service is any service that adds value to any other service/resource called its goal*. An example of an enhance service is a service that advertises another service, or manages it. Since we derive all the service patterns based on structures which were described in chapter 4, first we demonstrate the pattern structure and the specification for enhance service with more detail. Figure 4-8 shows the pattern structure for the enhance service. The left side of the structure shows the enhance service and its stock flow relationships with intentional and physical resources. The right side of the structure shows any kind of service with its relationships to the resources. The relationship between enhance service and the other service is "*enhance". For instance, the service in the left side can be a management service and other side can be an operational service. One can argue that the possibility of using "part of" relationship instead of "enhance". But in our view, the "enhance" relationship is a special kind of relationship. The objective of enhance is improve the efficiency or the effectiveness of affecting service / resource, like a supporting activity. And the affecting service can exist independently without an enhance relation. So, we differentiate *enhance* relationship from the *part-of* the relationship.

According the constraints of the specification of enhance service pattern structure have stock flow relationship with intentional resources. The chance of using operational resources is also high. As enhance service is a kind of service itself, it may use other services or it may have sub-service or it may be enhanced by other service. But these are optional. Therefore we exclude those from the pattern structure specification (cf. 4.1.5).

We identify several categories of enhance services as follows. Separation of operational layer and the enhance layer provides better insight to each of the services given below. The detailed analysis of these services is given in next sub sections.

- management service
- human resource provisioning service
- publication service
- access as a service

5.2 Management as a Service

The main objective of management is getting maximum results with minimum effort and minimum resources. Successful management is a key factor for the success of the business. Hence, businesses pay lots of effort to manage its activities. Porter defined management activities under *Firm Infrastructure* which is one category of the supporting activity. *Firm Infrastructure* consists of various activities including management activities such as general management, planning and quality management. *Firm infrastructure* supports entire value chain in the firm. As the goal of support activities is increase the efficiency and effectiveness, the *firm Infrastructure* also has the same goal. In service modeling approach, we also identify management activities separately. As the management involves planning, monitoring and evaluation activities, it cannot be categorized as physical resources or intentional resources. So the management is a service, which uses resources. These include operational resources and intentional resources. The management service may have sub-services too. The goal of management service is another service or resource. Hence, we have conceptualized management as an enhance service. In Software Engineering, the idea of separating operational and management concerns is not new. In the field of self-adaptive software, a distinction is made between internal and external adaptation (Nayak et al., 2007). Internal approaches intertwine application and adaptation logic. This has certain drawbacks. External approaches use an external adaptation engine or manager that contains the adaptation logic, the other part being called the "adaptable software". By conceptualizing management as a service, we follow an external approach. Note that this is not a formal necessity but an architectural choice.

We propose a *fractal* design approach in the sense that the same generic service model is applied to management services. We identified management service is itself another service that uses resources. These include operational resources, e.g., labour hours, and intentional resources. The management service may have sub-services, such as a monitoring service, and a management sub-service may have its own manager service. This service-oriented approach increases reusability. The advantage of the fractal approach is that it makes the design completely service-oriented, not only its operational part. This is in contrast to other approaches that for instance conceptualize BDI agents as management services, or monolithic BAM software.

What does a management service actually do? Most current approaches in the field of self-adaptive software follow classical control theory and posit a control loop, also called MAPE cycle (Monitor-Analyze-Plan-Execute). The same cycle underlies the deliberation cycle that is used in multi-agent systems (with Beliefs, Desires and Intentions). Our approach is required to be business-driven and service-oriented. Following the management control literature Simons (2000) we call it the diagnostic control cycle.

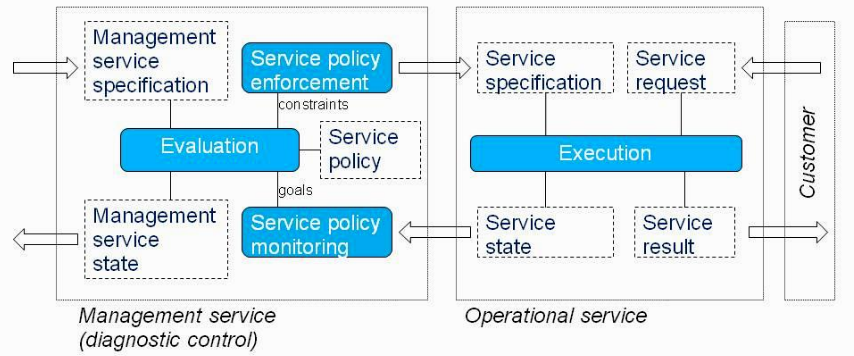

Figure 5-1: Diagnostic control cycle together with the service interaction cycle

(Dashed rectangles indicate intentional resources, coloured boxes indicate services)

Figure 5-1 depicts the generic service-oriented management architecture for the diagnostic control cycle. On the right hand side, we see the traditional service interaction cycle: the customer sends a request to the service provider. The execution produces, perhaps iteratively, a certain state that corresponds to and so fulfills the request, and this result is returned to the customer for evaluation. However, the execution does more than that. From a management perspective, the execution is the realization of the service specification. So there is another interaction cycle, between management service and operational service: the manager enforces a *service policy* on the operational service. In the case of software services, the *service specification* may take the form of a BPEL specification, or a set of business rules (cf. Moscinat et al., 2010; Yu et al., 2010) for how to implement policy enforcement based on such models). The execution produces a certain state (set of *assertive* – this reporting is also governed by policy constraints). The state information is returned to the manager, where it is typically aggregated by monitoring services and then evaluated. If the evaluation is not satisfactory (does not match the policy *goals*), the service specification is adapted. The policy will usually contain conditional constraints that become effective in the case of contingencies, akin to terms in a contract or a mitigation plan.

It is possible that the operational service policy has to evolve itself. However, this is not the responsibility of the management service. If we want this type of self-adaptation, a second management level has to be introduced, in accordance with our fractal design principle. In that case, the Management Service specification in Figure 5-2 is not fixed but itself the result of a Management Policy enforcement process.

Three kinds of management sub-services can be distinguished. A *monitoring* service uses and produces assertives. An *evaluation* service uses assertives and produces evaluatives. An *enforcement* service enforces policies, using evaluations and possibly assertives and producing directives. Further specializations are, for instance, sensor services, aggregation services, inference services and data transformation services as specific monitoring services.

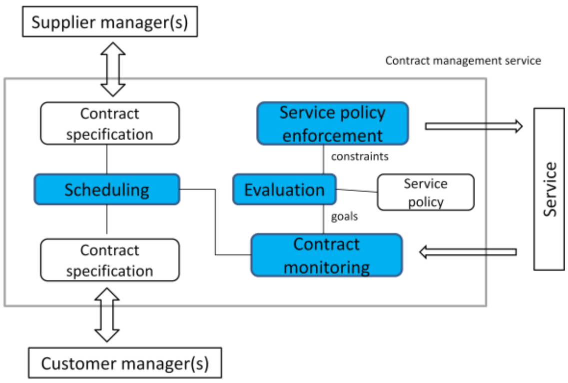

Figure 5-2: Contract management cycle

The flexibility within the service policy that enables varying enforcements can be realized in several ways that go beyond the scope of this research. We just want to mention two options. The policy may consist of a fixed set of alternatives, as in the variability transformations approach Cetina et al., (2009), that are selected on the basis of the evaluation. Or the policy contains a parameter whose value is dependent on the evaluation. An example is “credit level” in an order processing service. If the credit level is too low, the company losses because of non-payments. If the credit level is too high, the company misses sales opportunities. To find an optimal credit level and adapt if when the circumstances change the manager can (re-)calculate the parameter value by means of a stochastic optimization algorithm.

According to Simons (2000), diagnostic control is the “automated pilot” that allows the human manager to spend his time on other things, in particular interactive control. The way Simons presents this it is not a homogeneous group. An important subclass concerns interactions with service stakeholders about the service requirements. These requirements can be diverse, but include requirements on future capacity. These requirements are passed through the value chain in reverse direction, from customer (market demands) to sales and further on to production and procurement. To account for this kind of interactive control, we need another management cycle (Figure 5-2).

The manager interacts with other managers in the value chain at customer and supplier side. The customer manager's requests do not concern a particular service instance, but a certain state or quality of the service as such. For instance, that the service has a certain capacity at a specified time. This leads to certain *commitments* that are part of a REA *contract*. The synchronization of contracts is what is called scheduling. The purpose of a schedule is to make sure that for all services the needed resources are identified, as well as when they will be needed (Hruby, 2006). A schedule is a collection of increment and decrement commitments, as well as mitigation plans. The increment commitments indicate the availability of the service at some future time, or the availability of the resource produced by the service. Decrement commitments concern the resources (sub-services or resources produced by sub-services) needed to fulfill the increment commitments. These decrement commitments must be gained from the managers of the supplying sub-services. In our conceptualization, the schedule is not a separate entity but the combination of these contractual commitments. Note that the scheduling usually runs independently from the operational service. It only prevents the operational service to break down when the actual service requests come. However, the scheduling may influence the operational service. For instance, if the sub-service providers are not able to commit to the required resource capacity, this is forwarded as such via the contract monitoring, so that the service policy enforcement can pro-actively find and bind other suppliers.

A second important subclass of interactive control distinguished by Simons is the ongoing conversations on probing the assumptions underlying the diagnostic control settings. One of the manager's interactive control tasks is to adapt the service policy when its assumptions do not hold anymore or to anticipate such a break-down. This can be realized by a *discourse* between managers, akin to the above-mentioned knowledge plane (Dobson et al., 2010).

As mentioned earlier, more control systems could be distinguished – boundary systems and belief systems. Whether these can be realized as special cases of the other ones, or deserve to be identified independently, is a question for future research.

5.2.1 Example

The proposed management service model has been evaluated in a real world case study of wine production (S-Cube, 2009). According to the case description, the goal of the Wine Producer is to maximize his production in order to adapt the monitored market needs. During the wine producing process quality assurance plays a major role. The Quality Manager, the Agronomist who is an expert of a branch of agriculture which deals with field-crop production and soil management, and the Oenologist who is an expert in wine and wine production involved in this process. They have to observe the vineyard parameters and to react to critical conditions that may happen during the cultivation phase. The wine production case has major phases namely vineyard cultivation, harvesting, fermentation and wine distribution and selling.

Following the management service modeling approach the first step is modeling core business services at the operational level (Figure 5-3).

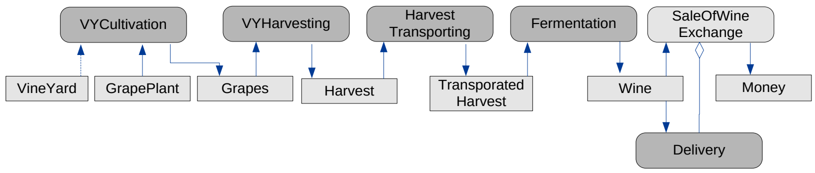

Figure 5-3: BSRM model for wine production (core services operational layer)

The core services of the wine production are vineyard cultivation, vineyard harvesting, harvest transporting, fermentation, and sales of wine. There is one exchange service which is sales of wine and all the others are conversion services. We use *ResourceExchange* and *Conversion* pattern structure specification. Vineyard cultivation is the first step of the wine production process. It is a conversion service and it uses resources Grape plants and Vineyard and produces Grapes. The next conversion service VYHarvesting uses the Grapes and produces Harvest. The rest of the core services use and produce resources are as depicted in Figure 5-3. The sale of wine which is an exchange service generates money in return for selling wine. To close the value cycle, the money derived from the sales of wine is spent on different activities in the wine producing process, for example to purchase grape plants.

Next, we focus on the vineyard cultivation core service and flowing Enhance pattern structure specification (Figure 5-4). We look for management services corresponding to the three control cycles. Vineyard cultivation management service *VYCultivationMgt* is a *contract management service* responsible for the cultivation process and the VY activity planning and labor allocation are sub-services to this management service. VY activity planning builds on contracts set up between *VYCultivation* and Sales, indirectly based on market information. VY quality management (*VYQualityMgt*.) service is a *diagnostic control service*. It is possible to identify two sub-services to VYQualityMgt, namely VY activity monitoring and recovery management. It uses a number of intentional resources as input and produces a service *policy* in the form of a recovery action list. Climatic data is an example of assertives that are defined as *policy assumptions*. To acquire these assertives, the management service presumably relies on a *discourse* (not included in the model). VY parameters are assertives for *monitoring* and the critical condition list represents *values* that support the *evaluation*. It turns out that the three control cycles and their sub-services provide a very good framework to structure and integrate the VY management phenomena.

The BSRM model aims to provide a first graphical overview of the management services. For each management sub-service identified, a more precise definition is to be made in a next step. This may be done using data models and data flow diagrams.

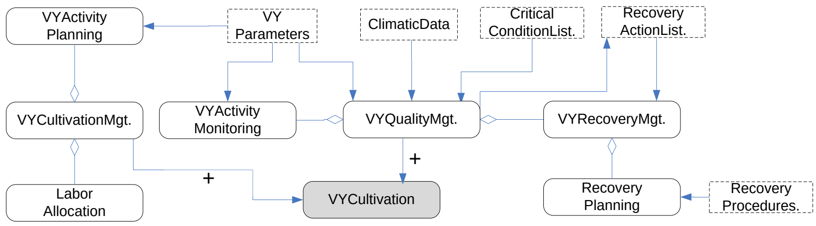

Figure 5-4: BSRM model for vineyard cultivation service (management layer)

5.3 Human Resource Provisioning as a Service

Human resource (HR) is not yet another resource used in the business's value creation process. In some companies HR is the main factor, to be in the competitive edge. For instance: software development industry. Human resource management is categorized as a supporting activity in Porter's value chain and HR management supports both primary and supporting activities. HR service provisioning is essential factor when realizing most of the exchange and conversion services. The questions are: how does the human resource differentiate from the other resources and how does the human resource provisioning affect to other services. As humans inherent with special qualities they are significantly different from other resources. They use their knowledge, skills, and experience, etc. when engaging in a value creation process. As a result of engaging the value creation process, humans are acquiring new knowledge, experience and motivation etc. The feelings of enthusiasm, urgency, intensity or the de-motivation towards the work (Macey et al., 2009; Schaufeli and Salanova, 2007) are also affected to the work that they are doing. They consume food and water to keep up their energy. All these attributes are coming from physical, cognitive and emotional aspects. These qualities are different from one to one. Hence, it is understandable that human resource is different from other resources. Coming back to the second question, one can view HR has a 'use' relationship with other services. But our view is HR engagement in a value creation process is more than that. HR involvement improves the efficiency and effectiveness of the affecting service. Porter also mentioned that all the supporting activities improve the efficiency and effectiveness of primary activities. Human resource service provisioning directly affect to the quality of the outcome of the service. Therefore, we model HR service provisioning as an enhance service.

Figure 5-5 shows the general pattern for any kind of HR service provisioning which enhances a conversion service. The general pattern is similar to the pattern described in section 4.5.1. The service which is affected by the *HRServiceProvisioning* can be any type of service (conversion, exchange, coordinate, enhance etc.) Bringing HR service to the company involves a cost always. Hruby demonstrated it as human resource acquiring pattern (Hruby, 2006). In the BSRM model, we capture the HR service acquiring as an exchange service. The company pays money to take HR service. In the figure, it shows as *HRServiceExchange takes HRServiceProvisioning* and *gives money* in return. Once HR service exists in the company, it enhances other service/s. In the generic pattern structure the enhance relationship shows between *HRServiceProvisioning* service and C*onversion* service. *HRServiceProvisioning* service uses and produces intentional resource/s. It uses/consumes physical resources as well. As *HRServiceProvisioning* is an enhance service, it follows the Enhance pattern structure specification. The constraints defined for the enhance pattern specification says that intentional resources are optional. But we further qualify this constraint for the HR service provision. As we mentioned at the beginning of this section, HR always uses and produces physical, cognitive and emotional features. Therefore HR service provisioning always has stock flow relationship (use and produce) with intentional resources which represent the human qualities (few of them are mentioned above). As these human qualities characterize the features of the knowledge dimension, the business service model shows the incorporation of knowledge dimension. We demonstrate the HR service provisioning with two examples in section 5.3.1.

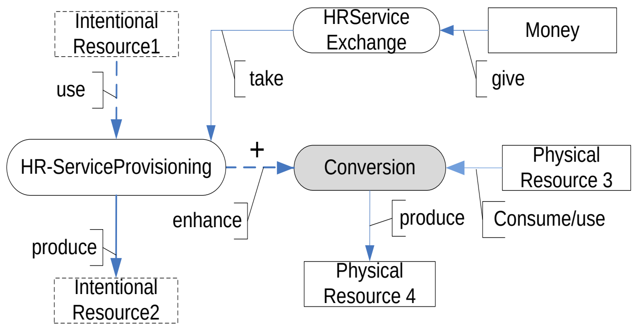

Figure 5-5: Generic Business Service pattern for the human resource provisioning

5.3.1 Examples

We demonstrate HR service provisioning with two examples. The first example is about painter's service provisioning (which relates to manufacturing sector). Figure 5-7 shows the BSRM model with minimal concepts which are sufficient to explain possible relationships. The company has to pay for the painter's service. It is modeled as *PainterServiceExchange* which *gives money* to *take* the *PaintingServiceProvisioning*. *Paint* is a conversion service which converts the unpainted product to painted product. The *Paint* service consumes the *Paint* resource. The painter's service helps to accomplish the painting. The model illustrates it as *PaintingServiceProvisioning* service *enhances* the *Paint* service .The *PaintingServiceProvisioning* and its relationships with resources are depicted in the left side of the figure. It uses and produces *PaintingSkill* and *Experience* as intentional resources. And also it uses *SafetyGlouse* – a physical resource, and consumes *Food* – physical resource. We use three pattern structure specifications to model this example namely exchange service pattern (cf.4.1.2), conversion service pattern (cf.4.12) and the enhance service pattern (cf.4.1.5). The BSRM model for *PaintingServiceProvisioning* points out conjunction of three dimensions namely salary administration as an exchange service, knowledge management as a HR service provisioning and operational process as a conversion service.

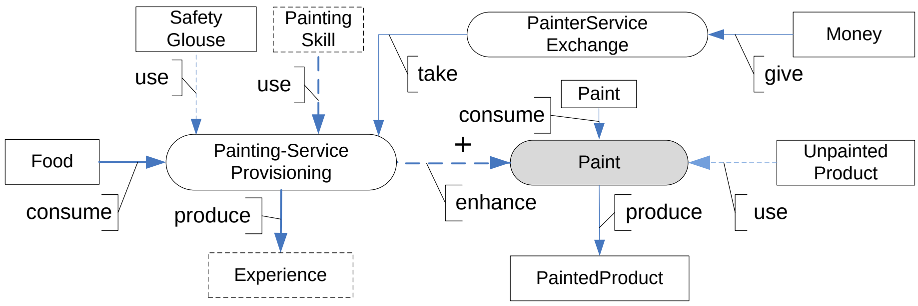

Figure 5-6: BSP for the painting service provisioning

The second example demonstrates the teaching service provisioning (which relates to service sector). *Teach* is the service which needs the teacher's service. In BRSM model (Figure 5-8), it shows as *TeachingServiceProvisioning* enhances the *Teach* service. Teacher's service is available in the system as a result of exchange service, which exchanges *TeacherService* and *Money*. *TeachingSkill*, *SubjectKnowledge* and *Experience* are intentional resources. The former two have *use* relationship and the later one has *produce* relationship with *TeachingServiceProvisioning.* Note that the *Learner* is an active role in this case. As the learner participates actively to the teaching activity, the *Teach* service is improved. Therefore, the role of learner is also modeled as an enhance service – *LearnerServiceProvisioning*. *LearnerServiceProvisioning uses LearningSkill* and *produces Knowledge. Teach* service has two enhance services from teacher's side and from the learners' side. *Teach* service uses *class room.* When there is a inactive role, we model it as a resource. For example the customer is an inactive role in a hair cutting service.

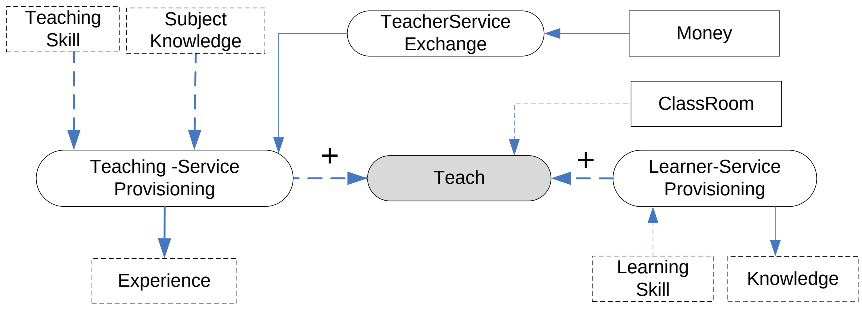

Figure 5-7: BSP for teaching service provisioning

We identified the HR service provisioning as an enhance service. The concept was demonstrated with two examples. In this discussion, we analyze our approach. The HR service provisioning in BSRM model gives an abstract view of knowledge management. We can identify several advantages by separating HR services provisioning with other services. Some of the advantages are highlighted in Businska and Kirikova (2011) research work, where they integrated knowledge dimension to business process modeling.

Benefits of separating HR service provisioning:

- Possibility to identify, plan, and manage knowledge of the role required for participating in a particular activity and linking this knowledge to the required service
- Possibility to evaluate the amount of lost organizational knowledge if a person – owner of knowledge – leaves the organization. i.e., to identify which tacit knowledge in this case should be transformed into explicit knowledge, such as documents, rules, systems, etc.
- Opportunity to improve understanding about the knowledge usefulness, validity and relevance for particular activities.
- Opportunity to enable competence requirements management and proactive training based on a process reengineering impact analysis.
- Opportunity to identify relevant costs to hire a personal.
- Opportunity to identify indirect costs well (Eg: Cost of food in the second example)

As an alternative modeling approach, we can consider HR as another type of physical resource which is used by a service. Then HR becomes a single component and special qualities which inherent by human being do not appear in the model. Then the knowledge dimension is not visible. Hence, the advantages, we discussed above, are also not evident.

If we look at the two examples demonstrated above, we can see, the BSRM model with HR service provisioning integrates several aspects. For instance in the product painting example, salary administration aspect, knowledge management aspect and operational activity are joint together.

5.4 Publication as a service

All kind of advertising activities relates to a product or service, are considered as publication service. A publication service provides information about another service (or any other resource) e.g., by means of a web page, a TV ad or a public service registry (Weigand et al., 2009). Companies spend huge amount of money for advertising their product/service. From a "budgetary" context perspective, the biggest part of marketing expenditures usually goes to advertising and promotion Ambler (2000). What does the publication service actually do? The primary objective of publication service is increasing the knowledge of customers about the product/service. It increases visibility of the product/service. In Porter's value chain analysis, advertising is bundled with sales and it is categorized as a primary activity. But, we distinguish sales and advertising as two different activities, where the sale is a primary activity and advertising is a supporting activity to selling product/service. In that sense, advertising service always has a goal of product/service. It always affect to a product/service. Hence, we categorize publication service as an enhance service.

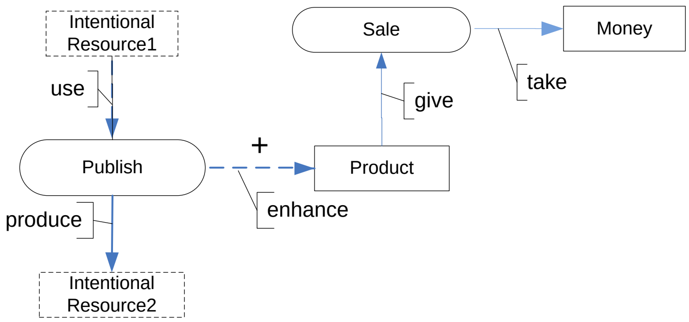

Figure 5-8: Generic business service pattern- Publication service

Figure 5-8shows the generic service pattern for publication service. *Publish* is an enhance service which enhances the *Product*. *Publish* service uses and produces intentional resources and it may use physical resources as well. The product/service which is enhanced by the *Publish* service is an input to the *Sale* service. As publishing is always incurred some cost, we demonstrated it by making relationship (*give*) with *money*. Further, *Publish* service may have *enhance* or *coordination* services. We didn't incorporate those services into the model as those are optional.

Publication as a service

5.4.1 Example

We demonstrate the publication service using an example – *Advertise* service. Figure 5-9 shows the service model for the *Advertise* service. The goal of advertising is making public awareness by providing information about the product. Hence, information of the product (*ProductInfo.*) and *ProductAwareness* are used and produced by the *Advertise* service. Further, the Advertise service uses *DigitalBoard* which is a physical resource. Normally, advertising and sales are tied together. Because the aim of the commercial advertising is encourage or persuade customers to purchase the product or service. In Porters value chain activities, these two are considered as one category in primary activity. We exhibit this relationship via the *Product*. The *Product* which is enhanced by *Advertise* service, relates with *Sale* exchange service with *Give* relationship. But sales and advertising are two separate services. The company gets *Money* in return from *Sale* exchange service. Companies can advertise their product by themselves or they can be outsourced the advertising service. If it is outsourced, the advertise service is inflow of an exchange service (advertise exchange service). Hruby (2006), discussed the same example under marketing and advertising. According him, *advertising service* is available by acquisition.

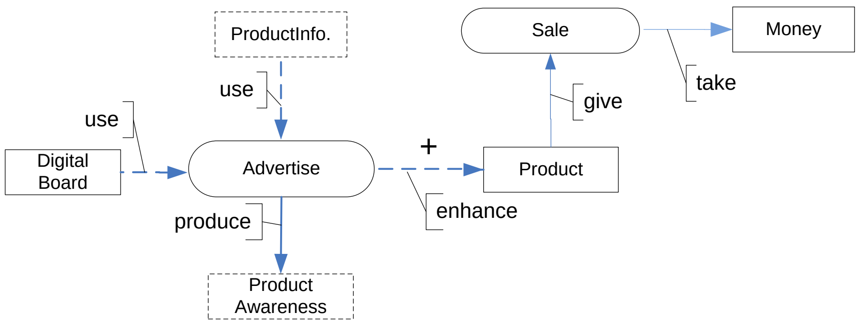

Figure 5-9: Business service pattern – *Advertise* service

(BSP-Advertise_Service)

In this section, we described the publication service as an enhance service. The pattern (Figure 5-9) is built using enhance pattern structure and the exchange pattern structure. The affiliation of sales and publication is represented by merging *Publish* service with *Sales* service via the *Product* (the common object). The separation of publication as a separate service has several advantages. Some of them are discussed in (Chesbrough, 2011).

Benefits of separating Publication service:

- Opportunity to identify relevant costs to for publicity.
- Opportunity to improve the publicity by analysing the customer perceptions.

- Opportunity to analysis and improve the HR services provisioning which enhances the Publication service. For example *Advertise* service may need advertising personal. The knowledge dimension of those personal, training of those personal , wages can be identified separately,
- Improves the loose coupling by separating publication service with other service, Eg: *Sales* and *Advertise* are two separate services,
- As publication service is separated, outsourcing the publication service is possible and it doesn't change the whole model. (cf. outsourcing pattern structure)

5.5 Access as a service

As more and more businesses move into the internet, the need of accessing them becomes vital. The general meaning of "Accessing" is entering or approaching. In SOA paradigms, we can find many definitions to the term service, but definition for access a service is lacking. However, in OASIS reference model (OASIS, 2006), the definition of service incorporates the two terms *access* and the *service*. They defined a service as *"a mechanism to enable access to a set of one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description*". We can find three characteristics of *Access* from this definition: access allows to reach to one or more capabilities; access has prescribed interface; it is running on constraints and policies of the service description. We use the same characteristics to define the *access* as a service. Apart from that, we incorporate the following characteristic too. If A is an access service to B, the following must hold: the goal of B is included in the goal of A, and B is a core service (Weigand et al., 2009). From business service point of view, access as a service uses resources (physical or intentional). Hence, we conceptualize *access* as an enhance service.

Figure 5-10 demonstrates a generic pattern for access service. *Access* service always enhances another service. It uses and produces *intentional* and *physical resources*. Like any others service, the *Access* service uses other services. It is possible to have enhance services to *Access* service. As they are optional, we exclude those services from the generic pattern.

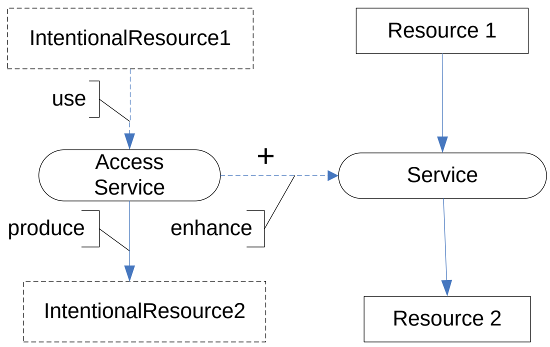

Figure 5-10: Generic business service pattern- Access service

Access as a service

5.5.1 Example

In the Figure 5-11, we demonstrate an access service to online book selling service called *BookSaleExchange. BookSaleExchange* is the core service which follows the exchange pattern structure. *BookSaleExchange* gives *Book* in return of *Money*. As book selling service is available online, there must be an interface to get into the service. Hence we add an access service (*BookSaleAccess*). *BookSaleAcces* service uses *WebServerSpace.* It uses *BookInformation* produces *convenience,* which are intentional resources. Since convenience is a strategic second order value, it is important to offer one or more user-friendly interfaces. For example, an order-by-phone interface. *BookSaleAccess* service is also enhanced by informational services. Examples are: product catalogue service (not in the figure).

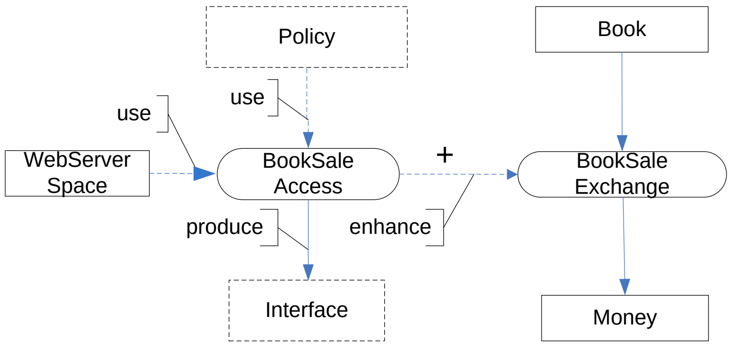

Figure 5-11: Business service pattern – *Access* service

(BSP-Access_Service)

We identified *Access* service is another kind of enhance service. Hence it follows the enhance pattern specification structure. However, the separation of access service has several advantages which are listed below.

Benefits of separating Access service:

- An advantage of using an access service is that it can act like Facade object in Software Engineering (Gamma et al., 1995) that induces loose coupling by hiding the service details from the consumer (Weigand et al., 2009).
- Improves the business agility. For instance, the core service can be access in different ways through access service which is convenience to its customers.
- Opportunity to analysis and improve the Access service.
- As the Accessing is a separate service it can contain medium-specific logic.

Chapter 6

Validation and Evaluation

6.1 Introduction

Validation and /or evaluation are a crucial component of a research project. The goal of validation is checking the internal and external validity of the research artifact. The purpose of evaluation is the systematic study of a research artifact in terms of functionality, completeness, consistency, accuracy, performance, reliability, usability etc. (Hevner et al., 2004). Among them, we check the completeness, accuracy and consistency of the proposed framework.

From the several methods of evaluation, we employ the following. Evaluation of the proposed framework has been described already in chapter 3 of the dissertation. Partial results were disseminated in the academic community such as conferences and publication in journals. The features of proposed BSRM language were compared with selected service modeling approaches and the results were discussed in chapter three. The logical consistency of the BSRM language was strengthened with a metamodel approach. As we derive the constructs and logic of BSRM in terms of a metamodel, the model can be evaluated in an objective way. BSRM has been formalized using the metamodeling facilities of ConceptBase (Jeusfeld et al., 2009).

In this chapter, we use three case studies to validate the proposed service design framework aiming to check its conceptual fitness and the completeness. We select three cases from the manufacturing domain, the agricultural domain and the service domain. First case study is about bike producing - Global Bike Inc. (GBI), a fictional case presented by SAP (Magel and Word, 2012). The next two are real world cases from literature. The second case is about wine producing published by S-Cube project (S-Cube, 2009). The last case is about a Dutch transportation company (Dieleman, 2010) which is an example in the service domain. The results of these case studies are described in section 6.2, 6.3 and 6.4 accordingly.

6.2 Case Study of Global Bike Incorporation

Global Bike Inc. (GBI) is a fictional case provided by SAP. GBI is a US registered specialized manufacturer and the seller of racing and touring bikes. It operates a subsidiary company - GBI Europe in Germany. US headquarters in Dallas is responsible for material planning, finance, administration, HR and IT functions apart from the manufacturing, warehousing and distributing. GBI Europe is conducting majority of R & D apart from the manufacturing, warehousing and distributing. GBI produces off-road bikes, touring bikes and accessories. Considering the size of the case, we select only one type of bike manufacturing i.e. off- road bike producing.

6.2.1 Business Service Model for the GBI Inc.

By following the design steps which are described in chapter 5, we start the modeling with enterprise service pattern of manufacturing company – *ESP_Manufacturing* (cf. Appendix C). The selected enterprise service pattern describes the main value activities of GBI Inc. in particular for off-road bike producing. Figure 6-1 shows the generic composite services which consists of purchasing raw materials, producing off-road bikes and selling bikes. We further zoom into the production service - '*Off-road Bike Produce*'.

GBI Inc. follows the make-to-stock scenario. Before starting the production, a production request has to be occurred and then the request becomes a planned order. Actual production starts after the planned order has been authorized into a production order. Issuing raw materials to the production and storing the finished good are two other activities which are required to complete the production. We drill down to actual production further. Assembling, inspecting, de-assembling and packing are sub activities of actual production. Following step 2, we decompose the '*Off-road Bike Produce*' service into sub-services and its related resources. We use the decomposed pattern structure to derive the decomposed model (Figure 6-3).

Step 1: ϕ → Enterprise model

In this step, we select the enterprise service pattern *ESP_Manufacturing*. Using pattern annotation, we derive the enterprise model.

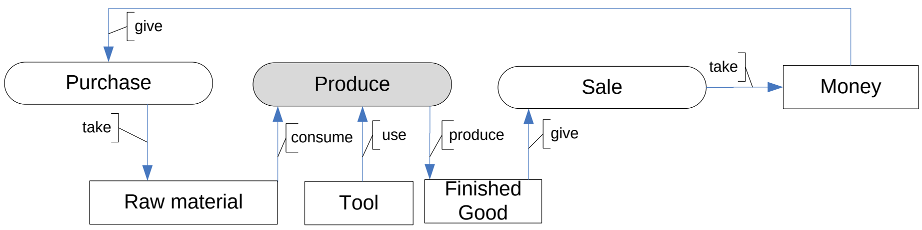

Figure 6-1: GBI Enterprise model for Off-road bike producing

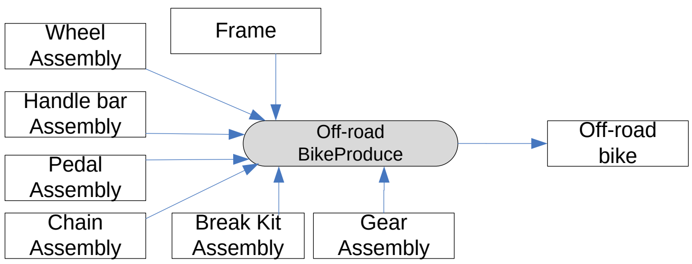

Figure 6-2: Bill of material to the Off-road bike producing

All the input materials of the '*Off-road Bike Produce*' service which are shown in Figure 6-2 are combined together as *Input Materials*, for the simplicity of the rest of the hereafter.

Step 2: Decomposition: $\{s_0\} \rightarrow \{s_0, s_1, _{\dots} s_n\}$, part-of$(s_i, s_0)$

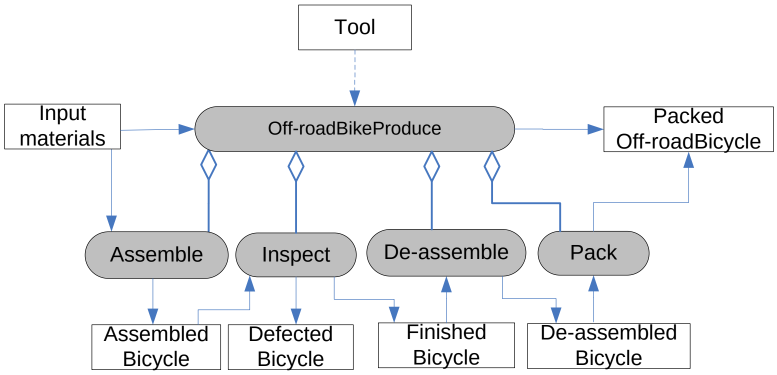

Figure 6-3: Decomposition of Off-road bike producing

Step 2 is the decomposition. We use Sub-Service (sub-service pattern structure) specification and two pattern operations (expansion and annotation) to derive the decomposed pattern (Figure 6-3). The patterns used in this step are *BSP-Inspect, BSP-Assemble. Assemble*. *Inspect*, *De-assemble* and *Pack* are conversion services. Note that the criteria for the decomposition levels are not worked out.

Step 3: Coordination: $\{s_{1,\dots,} s_n\} = \{ s_0, s_{1,\dots,} s_n, c_i,\}$, part-of$(c_{i,}, s_0)$, coordinate$(c_{i,}, s_0)$

Then we move to step 3 (Figure 6-4). As multiple sub- services are introduced, they need to be coordinated. We identify the production order processing as a coordination service -'*ProductionOrderProcessing'*. We use Coordinate specification and the pattern *BSP-Production_Order_Processing* to model the coordination service. *Production Order* which is an intentional resource is an input and also an output to the *ProductionOrderProcessing*. After accepting the production order, actual production is started. Coordination of different activities till the end of the production is done through *ProductionOrderProcessing*. Once the production is completed the "completed production order" is produced. In our model the different states of the production order (planned, accepted, issued and completed) are depicted. These states are specializations of the *ProductionOrder*.

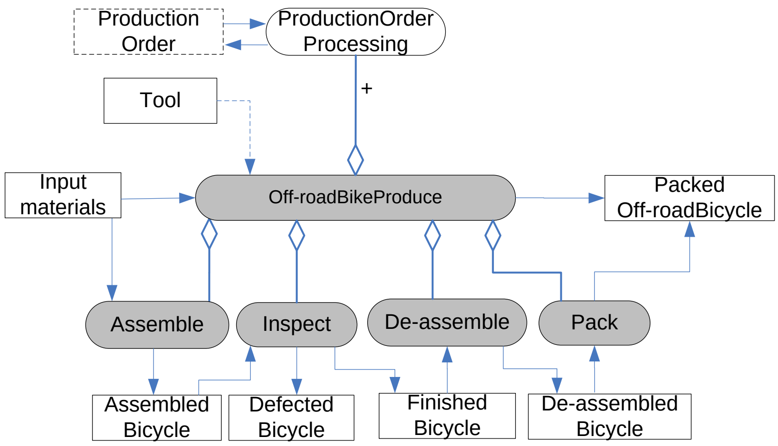

Figure 6-4: Introducing a coordination service

Step 4: enhance: {s} → {s,e} enhance(s,e)

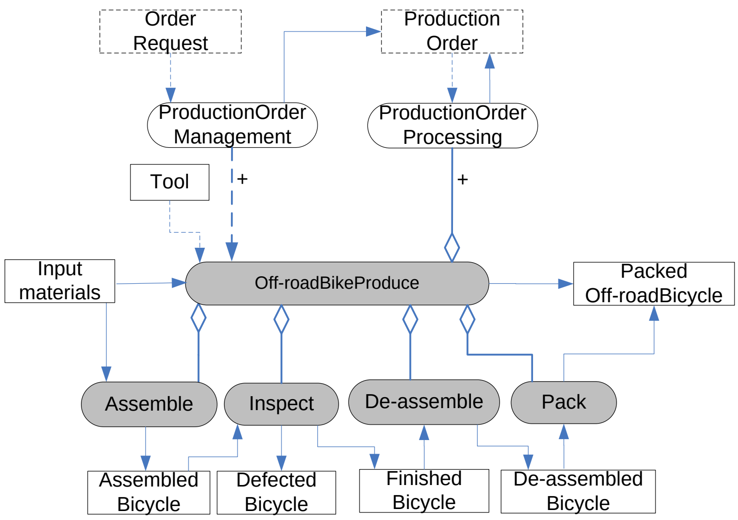

Figure 6-5: Introducing a management service

Proceeding to the next step, we identify - *ProductionOrder*Management as an enhance service (Figure 6-5). Planning of the production order, authoring it and monitoring the actual production are the main activities of *ProductionOrder*Management*.* We use Enhance specification and the *BSP_Production_Order_Mgt*. pattern in this step. It is possible to decompose these sub–activities (Figure 6-6) using Sub-Service specification.

ProductionOrderPlanning use *OrderRequest* and it generates *PlannedOrder.* The *ProductionOrderAuthorizing* approves the *PlannedOder*, and then the order becomes a *ProductionOrder* which is used by the *ProductionOrderProcessing*.

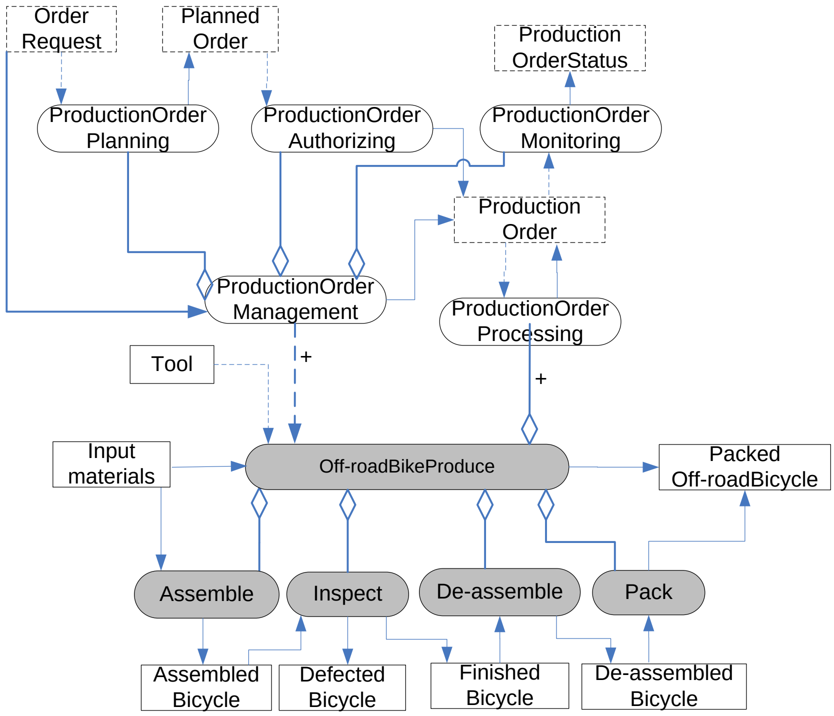

Figure 6-6: Decomposition of Production Order Mgt.

Step 5: Extension: $\{s_1, s_2\} \rightarrow$ merge(s_1, s_2)
Merging of Off-road bike producing with inventory of raw material.

We extend the model by merging inventory of raw material with off-road bike producing (Figure 6-7). The model of inventory of raw material can be derived from the pattern *BSP- Inventory_Raw_Material*. Then the model can be extended by merging the *IssuedMaterial* with *Off-roadBikeProduce*. The diagram given below shows the outcome of this step.

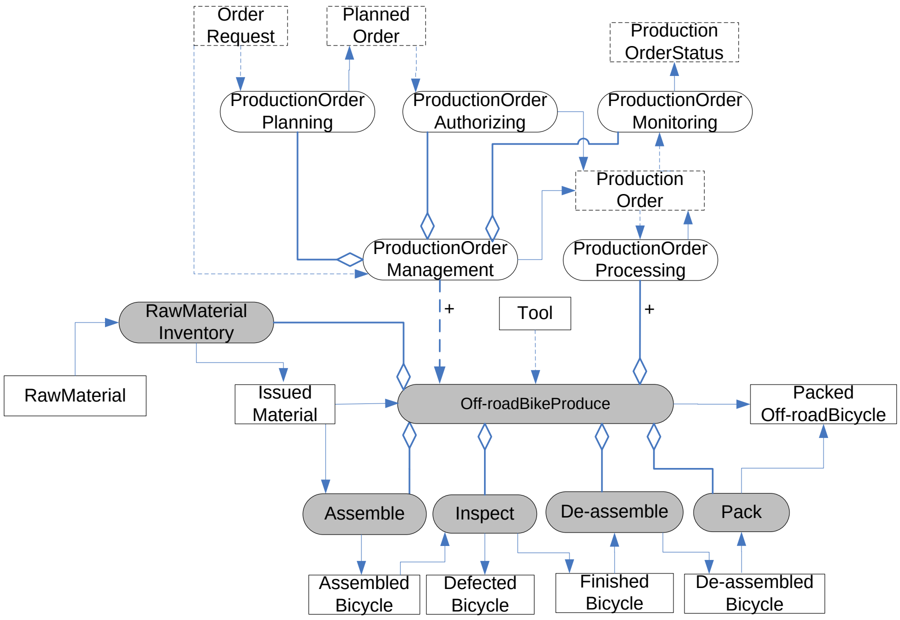

Figure 6-7: Merging of Off-road bike producing with inventory of raw material

We can repeat the steps 2 to 6 to capture the full details of the case. For example the conversion service *Assemble*, can be decomposed further according to the case description. We skip the design step 6 – the specialization as it not visible in this case at this moment. Therefore, we move to step 7.

Step 7: outsource, reengineer

GBI Company assembles only the wheels in-house and purchases other parts as assembled components. This is an example of outsourcing a product. For the simplicity of the diagram, we select the purchasing of tires for off-road bike. Following the pattern *BSP-Cash_Purchase,* we model the purchasing of tires. The diagram 6-8 shows the above detail. For the sake of clarity, we include only a part of the picture.

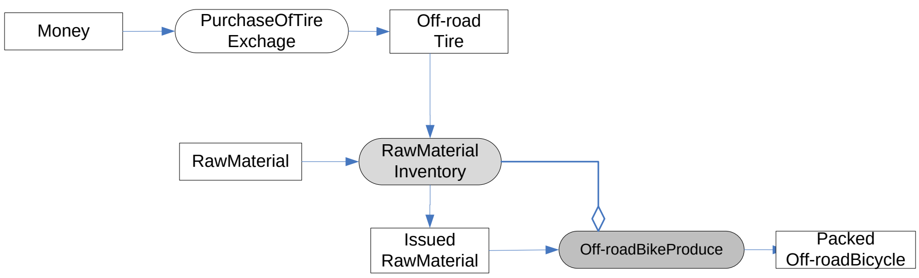

Figure 6-8: Purchasing of tires

6.2.2 Model Analysis

In this section we check the completeness of the BSRM model of GBI Inc. and the usage of business service patterns.

Model Completeness:

Figure 6-7 shows the BSRM model only for the Off-road bike producing. All the services are shown in rounded rectangles and the resources are shown in rectangles. Following the BSRM language semantics (cf. 3.1.1), we check the completeness of the model (Figure 6-7) with following features.

1. *Service stockflow completeness:*

Requirement: For each service, at least one inflow and outflow resource should be specified.

Checking: All the services which are depicted in colored rounded rectangles have at least one inflow and outflow resource.

2. *Physical resource flow completeness:*

Requirement: For each internal physical resource type, at least one inflow and outflow service should be specified. For external ones, this is not required.

Checking: All the physical resources are depicted with rectangles in the figure. We can observe the several resources do not meet the requirement. These resources are *RawMaterial*, *DefectedBicycle, Tool, PackedOff-roadBicycle*. The *RawMaterial* has only outflow service. The inflow service can be introduced by merging purchasing of raw material service. The *DefectedBicycle* has only inflow service. The possible outflow service can be modeled using waste handling. As we didn't modeled the procurement activities for all the infrastructure resources, the resource *Tool* has only outflow service. It can be replenished by connecting to the inventory handling of supporting resources. The *PackedOff-roadBicycle* has only inflow service. The sales exchange service is used the *PackedOff-roadBicycle* in the complete model.

.

3. *Enhancement and coordination completeness:*

Requirement: For each enhancement and coordination, at least one inflow and one outflow of the intentional resource type should be specified.

Checking: All the services which are depicted in white color rounded rectangles have at least one inflow and outflow intentional resources.

4. Intentional resource completeness

Requirement:	For each internal intentional resource type, at least one inflow and outflow service (enhancement or coordination) should be specified.
Checking:	All the intentional resources are depicted with dashed rectangles in the figure. All of the intentional resources meet the above requirement except the *OrderRequest* and the *ProductionOrderStatus*. As the GBI Inc. follows the make-to-order scenario, the *OrderRequest* should be generated by inventory of finished goods. *ProductionOrderStatus* has only inflow service and it can be used by a monitoring activity in the finished goods inventory in the complete model.

Usage of BSPs:

If we analyze the final BSRM model of the GBI Inc. case, we can see the usage of business service patterns. In step 1, we use the enterprise service pattern for manufacturing company. In step 2, we use Sub-Service specification and two patterns; BSP-*Assemble* and *BSP-Inspect*. Apart from that, in total we use *BSP-Production_Order_Processing*, *BSP-Production_Order_Mgt*. *BSP-Inventory_Raw_Material and BSP-Cash_Purchase* patterns in the solution. We can observe from the Figure 6-7, the large amounts of concepts are built on patterns. The *Merge* operator is also used when combining patterns.

6.3 Case Study of Italian Wine production

We continue the case study of wine production (S-Cube, 2009) that we used in the chapter 5, as the second demonstration. In chapter 5, we mainly focused to the management layer of the design. According to the case description, the goal of the Wine Producer is to maximize the production and the quality of wine in order to adapt the monitored market needs. The case reported wine production activities using four main scenarios, i.e. managing market needs, cultivation handling, harvesting and fermentation and distribution and selling. Different personals are responsible for activities of the wine production. For example the Quality Manager, the Agronomist who is an expert of a branch of agriculture which deals with field-crop production and soil management, and the Oenologist who is an expert in wine and wine production involved in this process. Each of the scenarios is briefly described in the next paragraphs.

Cultivation handling: The cultivation starts with determining the kind of vineyards to be cultivated by analyzing market data. The responsible personals for vineyard cultivation have to observe the vineyard parameters, detect the critical conditions and perfume the recovery actions during the cultivation phase.

Managing market needs: In this scenario the main activity is analyzing market data, forecast, observe the vineyard activity and react for the critical conditions.

Harvesting and Fermentation: One main objective of this phase is minimizing the time interval between harvesting and grape processing. Evaluating the climatic information for harvesting, transporting the harvest, during the fermentation and the store is essential.

Distribution and sale: During this phase order processing, delivering and goods-return are the main activities. Controlling the relevant climatic parameters during the delivery is essential.

6.3.1 BSRM for Italian Wine production

We start the BSRM model for the vineyard case study following the patterns together with the design steps described in chapter 5. There is no strict chronological order to the design steps except 1, 2 and 3.

Step 1*: Ø → Enterprise model*

We derive the enterprise model for the wine production annotating the enterprise service pattern *ESP_Manufacturing*. Figure 6-9 shows the initial business service model. The main conversion services are purchasing plants and wine production. The main exchange service is selling wine (*SaleOfWineExchange*).

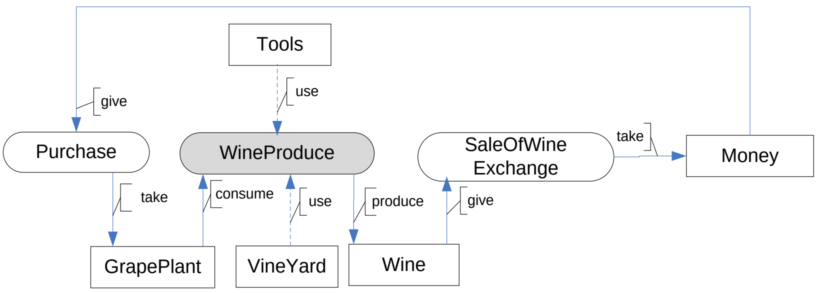

Figure 6-9: BSRM enterprise model for wine production

Step 2*: Decomposition $\{s_0\}$ → $\{s_0, s_1, ..., s_n\}$, part-of(s_i, s_0)*

We zoom into the *wine produce* and *sale* services. The production of Wine has multiple activities. In this step we decompose the *WineProduce* core service (Figure 6-10) and the Sales (Figure 6-11).

The core-sub services of the wine production are *vineyard cultivation*, *vineyard harvesting*, *harvest transporting* and *fermentation*. All of these are conversion services. Using Sub-Service pattern structure specification, the model derived in the previous step is decomposed. The *VineyardCultivation* is the first step of the wine production process. It is a conversion service and uses the *Grape Plants* and the *Vineyard* resources. The *VineyardCultivation* produces the *Grapes*. The next conversion service *VYHarvesting* uses the *Grapes* and produces the *Harvest*. The rest of the core-sub services *use* and *produce* resources as depicted in Figure 6-10.

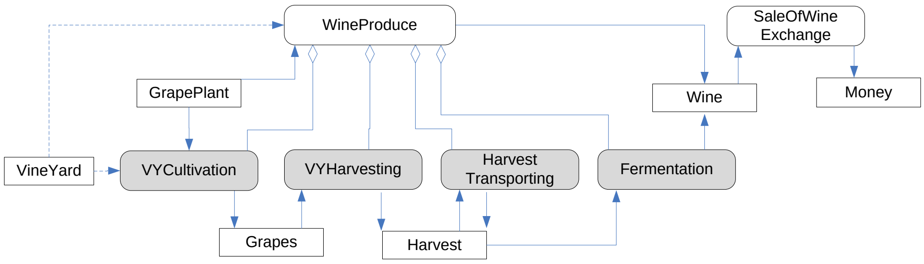

Figure 6-10: Decomposition of *WineProduce* service

The sale of wine which is an exchange service generates *money* in return of selling *wine*. The sale of wine is modeled following the ECX-P (pattern structure specification for sale of physical resource). To close the value cycle, the money derived from the sales of wine is spent on different activities in the wine producing process, for example to purchase grape plants. *IssueWine* and the *ReturnOfWine* are sub-services of *SaleOfWine* (Figure 6-11). The *IssueWine* represents the services regards with issuing wine for the sale and it follows the *BSP-Issue_Final_Product* pattern. The *ReturnOfWineExchange* describes the activities of returning wine and it follows the *BSP-Product_Return pattern*. Here after we focus only into to the Sale of wine.

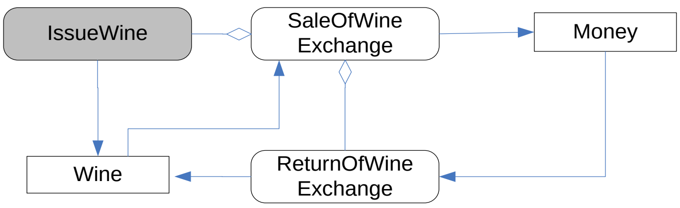

Figure 6-11: Sub-services of sale of wine

Step 3: *coordination:* $\{s_{1,\ldots,}\, s_n\} = \{\, s_0\, , s_{1,\ldots,}\, s_n,\;\; c_i,\}$*, part-of*$(c_{i,,}\, s_0)$*, coordinate*$(c_{i,,}\, s_0)$

Once after introducing the sub-services, the next step is identifying the coordination services. One of the coordination services for *SaleOfWine* is *SalesOderProcessing,* which processes the activities of sales order. The intentional resources *SalesOrderRequisition* is used and *SalesOrder* is produced by this coordination service (Figure 6-12). We follow the *BSP_Sale_Order_Processing* pattern.

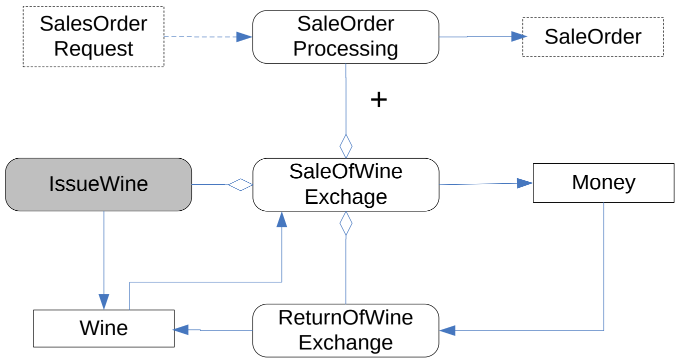

Figure 6-12: Coordination of sales of wine

***Step 4**: Outsourcing*

The wine producer uses an external delivery company to transport wine. Hence, it follows the outsourcing pattern. The *Delivery* service is used by *SaleOfWine* service. According to the constraint of Exchange_Outsource specification, the outsourced service should bring at least one resource from the service provider's side. In this case we depicted it as *Truck* (it is not described in the case). And also at least one internal resource should be affected by the outsourced *Delivery* service. In our case, it is *Wine*. Wine is added value by transporting it from one location to another. The company has to pay for the delivery service. In the model *DeliveryExchange* service represents the buying of delivery service.

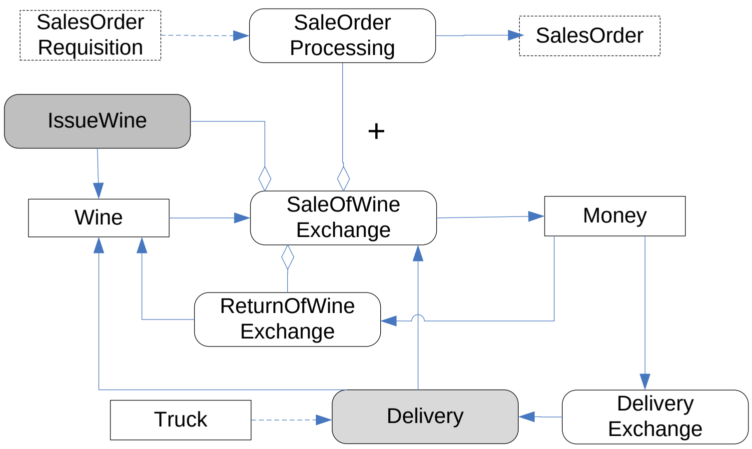

Figure 6-13: The delivery service outsourcing

Step 5: Enhance: {s} → {s,e} enhance(s,e)

In this step, we identify the enhance services. There are two main enhance services. One service enhances the *SaleOfWine* and the other enhances the Wine (Figure 6-14). The former is *MarketInfo.Mgt*. which manages the market information. It uses the market information and produces predicted sales information. The later is *WineQualityMgt which* is a diagnostic control service (Weigand et al., 2011). This involves the activities of monitoring the temperature, other climatic data and react to the critical conditions of the wine during the issuing and transporting. The quality checking of returned wine is also another important task of the quality management. The *WineQualityMgt* can be further decomposed to sub-services namely *Monitoring* and *RecoveryMgt*. , as depicted in the Figure 6-15.

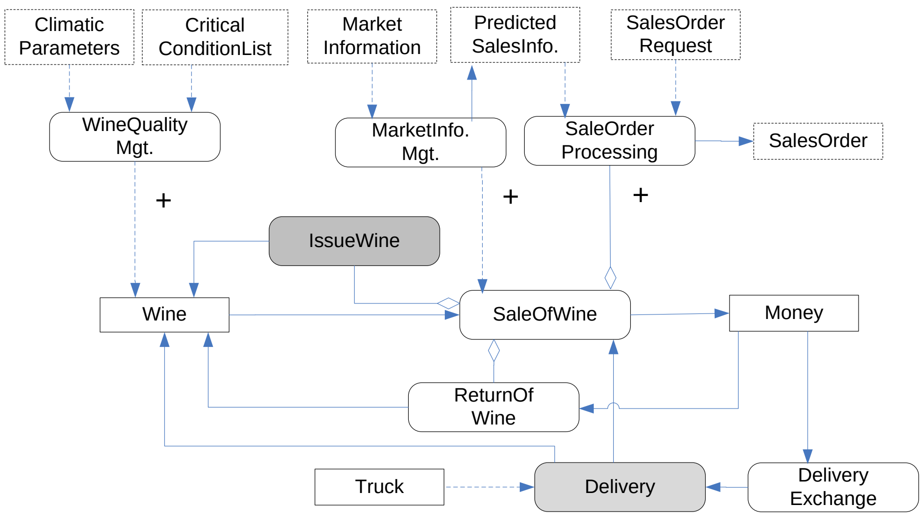

Figure 6-14: Enhance services for the Sale of wine and the wine

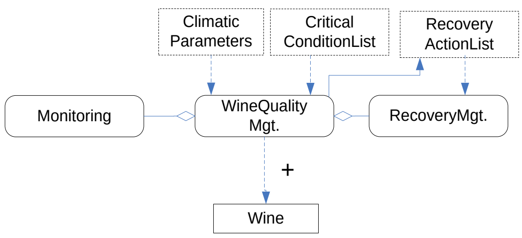

Figure 6-15: Decomposed *WineQualityMgt*.

6.3.2 Model Analysis

In this section, we check the completeness of the BSRM model for the wine production case in particular the sale of wine activity. In addition we analyze the usage of business service patterns it the model.

Model Completeness:

Figure 6-14 shows the BSRM model only for the sale of wine. All the services and the resources are shown in rounded rectangles and rectangles respectively. Following the BSRM language semantics (cf. 3.1.1), we check the completeness of the model (Figure 6-14) with following features.

1. *Service stockflow completeness:*

Requirement: For each service, at least one inflow and outflow resource should be specified.

Checking: All the services except enhance and coordination are depicted in white color rounded rectangles named with suffix word "Exchange" and the colored rounded rectangles. All of these services except *IssueWine* met the above requirement. The *IssueWine* service has only outflow resource. In the complete model of wine production case, we can identify an inflow resource by connecting the *IssueWine* service with the *InventoryOfWine* service.

2. *Physical resource flow completeness:*

Requirement: For each internal physical resource type, at least one inflow and outflow service should be specified. For external ones, this is not required.

Checking: All the physical resources are depicted in rectangles in the figure. We observe one resource doesn't have inflow service. That is *Truck*. As *Truck* belongs to external agent, modeling the inflow service is out of the scope.

3. *Enhancement and coordination completeness:*

Requirement: For each enhancement and coordination, at least one inflow and one outflow of the intentional resource type should be specified.

Checking: All the enhancement and coordination services have at least one inflow and outflow intentional resources.

4. *Intentional resource completeness*

Requirement: For each internal intentional resource type, at least one inflow and outflow service (enhancement or coordination) should be specified.

Checking: All the intentional resources are depicted in dashed rectangles in the figure. Several intentional resources are not aligned with the requirement. These are *ClimaticParameters, CriticalConditionList, SalesOrderRequest* and *SalesOrder*. The *ClimaticParameters* has only outflow service. The climatic data is obtained from the weather reporting. It can be generated by outside company, for example meteorology department. Then the inflow service is not in the boundary of the wine production case. The next incomplete intentional resource is *CriticalConditionList.* It has only outflow service. It is possible to define R & D service to generate such a *CriticalConditionList.* The *SalesOrderRequest* is generated by analyzing the market data. It is possible to define a service called *MarketDataAnalysis*. The last incomplete intentional resource is *SalesOrder*. If we decompose the *SaleOfWine*, we can distinguish payment as a sub-service. Then the *SalesOrder* is an inflow resource to the *Payment* service.

Usage of Patterns

We analyze the usage of business service patterns using the BSRM model of wine producing case. In step 1, we use the enterprise service pattern for manufacturing company called *ESP_Manufacturing.* We employ two patterns in step 2 namely *BSP-Issue_Final_Product* and *BSP_Product_Return.* In the next step *BSP_Sales_Order_Processing* pattern is used. Using Exchange_Outsource specification, delivery outsourcing is modeled, in the fourth step. Finally the management service is introduced to the mode using Enhance specification. All the services in Figure 6-14 are derived using pattern structure specification and /or business service patterns from the library.

6.4 Case Study of VDB – A Transport Company

We select the third case from logistic domain to represent an example from service sector. VDB (Dieleman, 2010) is an international logistic service provider. The company is specialized in transportation of dry and liquid bulk products by road. The total number of employees in the entire VDB Group is around 1200, among which there are around 900 truck drivers. The main goal of VDB is providing transportation to its customer's goods. The whole process involves customer order processing, allocating resources (trucks and drivers), and monitoring real transportation. In case of lack of resources, VDB hires trucks with drivers from other small companies in the area. The VDB Company has its own repairing center for vehicle repairing.

6.4.1 BSRM for VDB

We start the BSRM for the VDB case study following the design steps described in chapter 5.

***Step 1**: Ø → Enterprise model*

We derive the enterprise model for the VDB annotating the enterprise service pattern *ESP_Transport*. Figure 6-16 shows the business service model.

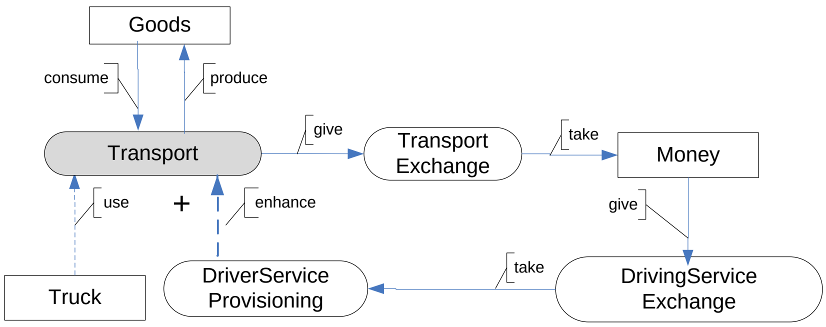

Figure 6-16: ESP-Transport for VDB

By definition, every commercial company has at least one exchange process, and this turns out to be the most central service in VDB as well. The *TransportExchange* is an exchange service that exchanges *transport* service in return for *money*. *Transport* is a service that is directed at *goods* of the customer and uses one resource type - the *Truck*. The drivers provide their service to the *Transport* service. We model it as *DriverServiceProvisioning* which enhances the *Transport* service. The *DriverServiceProvisioning* exists in the company by acquiring the driver's service. In the enterprise model, the acquiring of the driver's service is depicted as the *DrivingServiceAcquisition* – an exchange service. The company gives money to take driver's service. We zoom in to the *Transport* service. As there are no sub-services visible in the case for *Transport* service, we skip step 2and move to step3.

Step 3: coordination: $\{s_{1,\ldots,} s_n\} \rightarrow \{ s_0, s_{1,\ldots,} s_n, c_i,\}$, part-of($c_{i,,}$ s_0), coordinate($c_{i,,}$ s_0)

ResourceAllocation is a coordination service as it plans a transport service execution and allocates the required resources. It could use some sub-services responsible for a specific kind of resource: truck versus driver. We use Coordinate specification to model the *ResourceAllocation* (Figure 6-17).

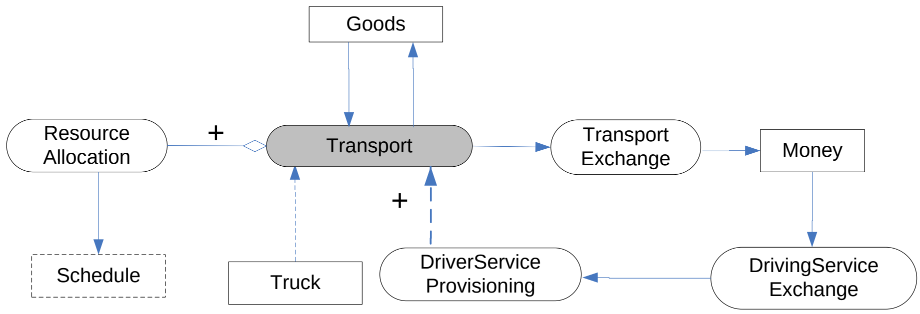

Figure 6-17: Coordinating the transport service

Step 3: enhance: {s} → {s,e} enhance(s,e)

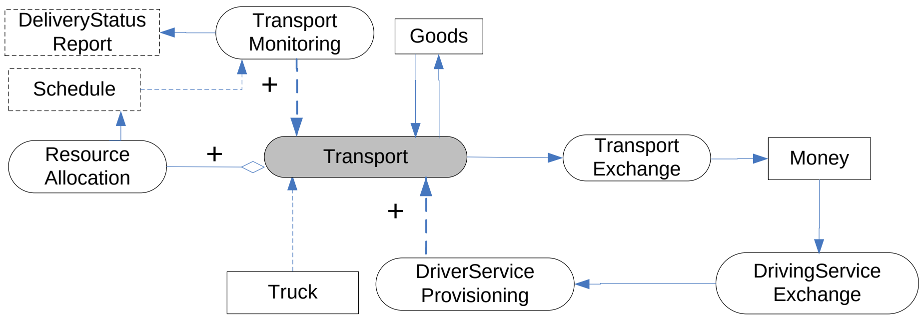

Figure 6-18: Enhance service for the transport service

We identify transport monitoring (in the figure it is named as *TransportMonitoring*) as an enhance service to the transport service. One objective of the transport monitoring is providing information about the delivery of goods. This information is depicted in the model as *DeliveryStatusReport* which is an intentional resource. We follow the Enhance specification to derive the model (Figure 6-18).

Step 4: outsource

According to the case description, the company is hiring trucks and drivers from a third party when the internal resources are not sufficient. Following the Exchange_Resource specification which describes the outsourcing a good, we model the hiring of trucks and drivers. Figure 6-19 shows the service model. *CharterExchange* is an exchange service which has 'give' and 'take' relationship with recourses (truck and driver) and money accordingly.

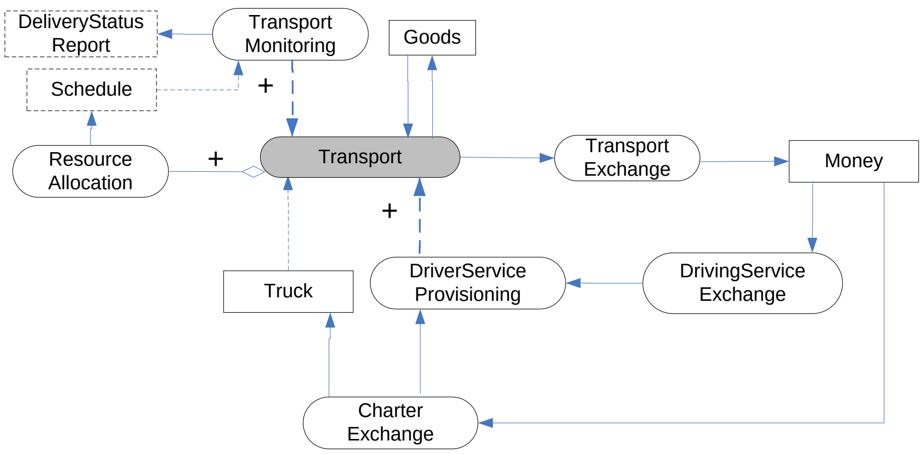

Figure 6-19: Outsourcing the trucks

Step 5: extension: $\{s_{1,} s_2\} \rightarrow$ merge $(s_{1,} s_2)$

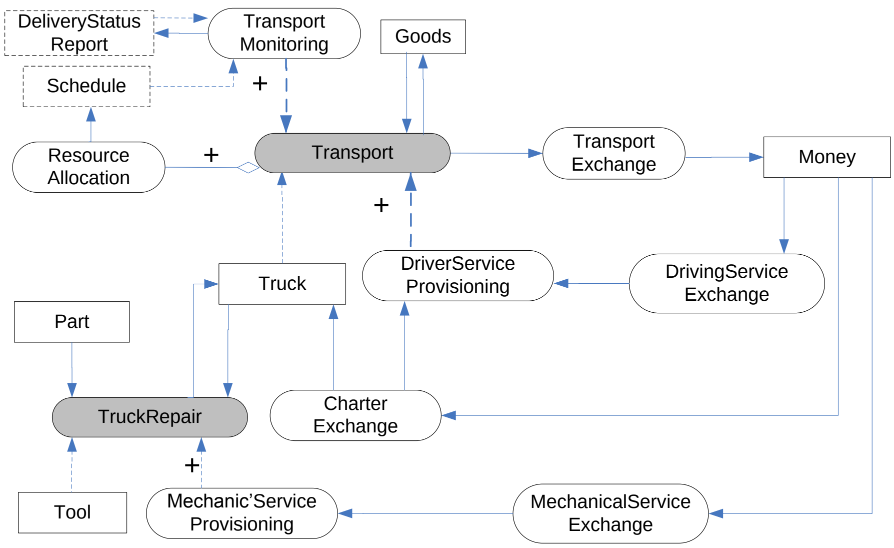

Figure 6-20: Merging the truck repair service

In this step, the model is extended. According to the case description, VDB has its own repairing centre for the *Truck* repairing. *TruckRepairing* is a conversion service that consumes and uses *Parts* and *Tools* accordingly. The repairing needs the service of mechanics to repair the trucks. Using the Conversion specification, we model the

TruckRepairing. The two models (model in the previous step and the *TruckRepairing* model) are combined using the *Merge* operator. Figure 6-19 shows the extended model.

6.4.2 Model Analysis

In this section, we check the completeness of the BSRM model for the VDB Transporten case in particular the transport activity. In addition we analyze the usage of business service patterns it the model.

Model Completeness:

Figure 6-20 shows the BSRM model for the part of VDB. All the services and the resources are shown in rounded rectangles and rectangles respectively. Following the BSRM language semantics (cf. 3.1.1), we check the completeness of the model (Figure 6-20) with following features.

1. *Service stockflow completeness:*

Requirement: For each service, at least one inflow and outflow resource / service should be specified.

Checking: All the exchange and conversion services have at least one inflow and outflow resource / service in the Figure 6-20.

2. *Physical resource flow completeness:*

Requirement: For each internal physical resource type, at least one inflow and outflow service should be specified. For external ones, this is not required.

Checking: All the physical resources are depicted in rectangles in the figure. We observe two resources do not meet the requirement. Those are *Part* and *Tool* and both have only outflow service. The *Part* can be connected with *Issue* service of the inventory of spare parts. The tools can be replenished by connecting to issue service from the inventory of tools

3. *Enhancement and coordination completeness:*

Requirement: For each enhancement and coordination at least one inflow and one outflow of the intentional resource type should be specified.

Checking: Several enhance services and coordination services not satisfy the above requirement. The driver and mechanic service provisioning enhance services can be further expanded using knowledge dimension features (cf. 5.3). If we model knowledge dimension features, we can define possible inflow and outflow intentional resources. The *ResourceAllocation* doesn't have inflow intentional resource. As the transport service is provided based on customer requests, the customer order is a possible inflow to the *ResourceAllocation*.

4. Intentional resource completeness

Requirement: For each internal intentional resource type, at least one inflow and outflow service (enhancement or coordination) should be specified.
Checking: All the intentional resources are depicted in dashed rectangles fulfill the above requirement.

Usage of Patterns

If we analyze the results of VDB case, we can see the usage of business service patterns. In step 1, we use the enterprise service pattern for the transport company. In step 2, we use Coordinate specification. Even though the pattern library provides BSPs for the manufacturing domain, applications of business service patterns are still possible in some situations. For example, in step 3 we introduce a management service called *TransportMonitoring*. The *TransportMonitoring* service follows the Enhance specification. The truck repairing activity is modeled in step 5. We use the *BSP_Repaire_Product* pattern to derive the *TruckRepair* service. Another advantage for the designer is flexible model extensibility. As we employed in step 5, the *Merge* operator facilitates the flexible extensibility for the model.

6.5 Discussion

We composed the business service model for the selected parts of three cases using the pattern structure specifications, the pattern structure operations, the patterns and the pattern operators. In each of the cases, we carried out the completeness checking of the results. We can observe that the service modeling framework is able to model the business services within the given case description. The incompleteness occurs due to the lack of details given in the case. In this section, we analyze the results from two perspectives. First, we discuss the added value for the designer. Secondly, we look at the BSRM model from business perspective.

6.5.1 Designers Perspective

We stated several goals of this research from the designer's perspective in chapter 1. In general, the designer of the software system is provided with a library of business service patterns together with pattern structure specifications, pattern structure operations and pattern operators to build the model. Further, he is guided with comprehensive design steps to start and compose the model. Modeling is an evolving process and a model is not a fixed picture which is built at one time. According to the new business requirement, the models need to be adjusted. The framework supports the designer for the flexible model extensibility. For example, a new service can be added to the existing model using the *merge* operator. The merge operator is one example which defines systematic merging of patterns / models. The model evolution includes outsourcing services. The outsourcing pattern structure specification allows bringing new services to the existing service model. The design steps support meaningful model evolution, rather than arbitrary deletions and insertions.

The business service patterns include a list of common abstract services, resources and their relationships. Hence, the designer is provided the domain concepts through the patterns. As the patterns represent generic view, the pattern annotation operation allows customize the pattern. The comparison of the results of three cases is given in Table 6-1.

Table 6-1: Comparison of the results

Futures	GBI Inc.	Wine Production	VDB
Sector	Manufacturing	Agriculture and Manufacturing	Service
Usage Enterprise Pattern	*ESP-Manufacturing*	*ESP-Manufacturing*	*ESP-Service*
Selected activity for the Model	Produce	Sales	Transport
Usage of Pattern structure specifications	Sub-Service Enhance Coordinate	Enhance Exchange_Outsource Exchange_Resource	Coordinate Enhance Conversion Exchange_ Outsource
Usage of business service patterns	*BSP-Assemble* *BSP-Inspect* *BSP-Production_Order_Processing* *BSP-Production_Order_Mgt.* *BSP-Inventory_Raw_Material* *BSP-Cash_Purchase*	*BSP-Issue_Final_Product* *BSP_Product_Return* *BSP_Sales_Order_ Processing*	*ESP-Transport* *BSP_ Repaire_Product*

6.5.2 Business Perspective

As we discussed in the previous three cases, the BSRM model helps to visualize the business as services for the business users. Hence, it improves the service-oriented thinking at business level by hiding the technical aspects. The notation helps to differentiate the different types of services and resources.

Separation of operational and management services:
The separation of operational activities from the supporting activities is modeled with BSRM. This separation helps taking necessary actions to improve operational activities. The separation also helps allocating separate budget Chesbrough, (2011), and to define organization policies to a particular activity which is represented by a service.

Flexibility:
The proposed framework provides a flexible model changes though merge operator and the outsourcing pattern structure specification. With business perspective, the flexible structure helps to visualise and analyse the services / products which can be outsourced.

Chapter 7

Conclusion and Future Work

In recent years, Service-Oriented Architecture (SOA) concepts have matured into an important architectural style for the enterprise information systems. Although the concept of service-oriented architectures (SOA) has been in discussion for several years, it is mainly regarded as a technological concept. As argued in section 1.2, a truly service-oriented design mechanism which incorporates the business level is an essential requirement to obtain the full potential of SOA. Therefore, the main focus of this research is establishing a service-oriented design mechanism which has a service perspective at business level. In this research, we propose a business service modeling framework which has a strong business modeling basis.

We summarize all the components (design science artifacts) of the business service modeling framework in Figure 7-1. In the figure, the existing concepts are denoted with filled rectangles and the new artifacts which are introduced in this research are denoted with white rectangles. We use circles to represent the combination of several concepts. The arrows show how the concepts are linked. The main constructs of BSRM and their relationships, are defined in the service metamodel which is grounded on well established business ontology – REA. Then we introduce the BSRM language based on the definitions in the service metamodel. The language is represented with a graphical modeling notation called BSRM notation. The service metamodel, BSRM language and the notation are discussed in Chapter 3. The next major step of the proposed framework is business service patterns. All the patterns are defined based on the specifications of service pattern structures. As the pattern structures are generic skeletons, we introduced two pattern structure operations (expansion and annotation) to generate domain specific patterns. When composing pattern, we use pattern operators (*Merge*) as well. New patterns can be derived by composing patterns recursively. Chapter 4 covers all the specifications of pattern structures, pattern structure operations and business service pattern operators and finally the business service patterns. Then we describe the BSRM mapping with conceptual data model, e^3-value network and the business process model. The first two mappings are discussed in chapter 3 and the last one is discussed in chapter 4. The integration of business service patterns with web services are also discussed in chapter 4. Finally, the systematic way of composing enterprise service model using business service patterns and design steps is presented in this dissertation.

This research work fulfills the need of a service oriented view at business level. The proposed framework provides a systematic way to model a business as services. BSRM, we claim to be the first specific service modeling language at CIM level. Both the designer and the business user benefit from the proposed framework. For the designer, the BSRM model gives an abstract layer of services to start the next layers of design. For example, by means of model mapping between BSRM and BPMN.

Further, the designer is guided with domain concepts by means of patterns. The framework facilitates to build the enterprise service model by following the design steps together with business service patterns, pattern operations and pattern operators. The proposed business service models have flexibility to future growth. The proposed service design framework is not limited to design the business service model for an enterprise. The framework supports web service integration. Some of the components of the service integration metamodel are shown in the picture with dotted line.

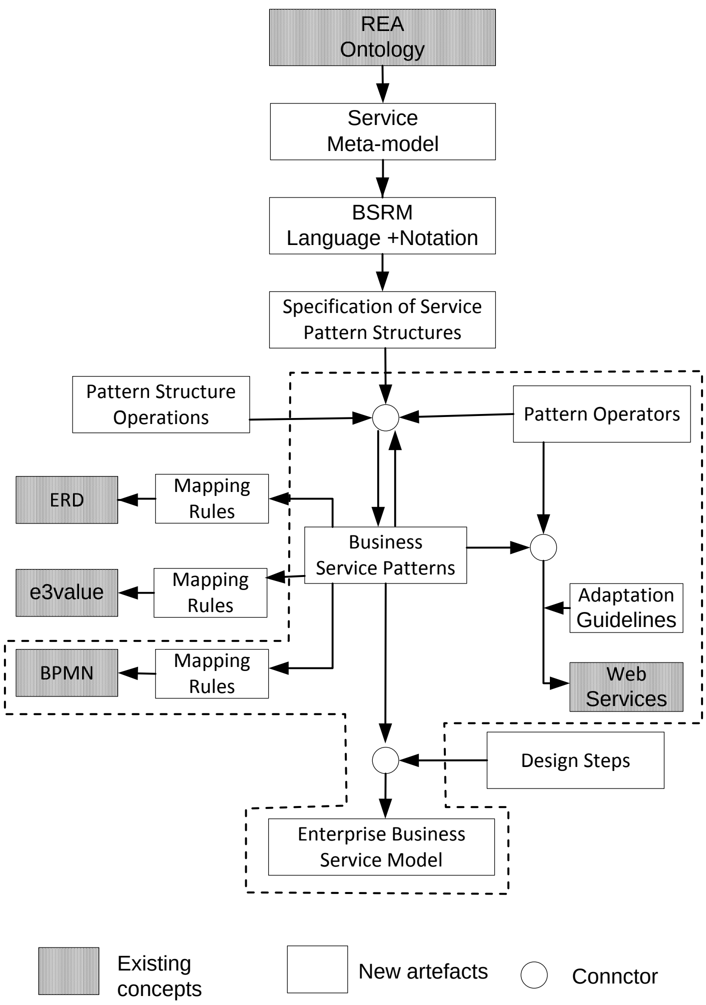

Figure 7-1: Summery of the business service design framework

For business users, BSRM model helps to visualize the business as services. Hence, it improves the service-oriented thinking at business level by hiding the technical aspects. In BSRM, the separation of operational activities from other activities through service classification is an advantage. This separation helps to successful business decisions. BSRM models can be used for business analysis, although in this research we could only give indications for that.

7.1 Research Questions and Answers

In chapter 1, we defined several research questions. In this section, we summarize the answers to those research questions.

1. *What is the State of the Art in service-oriented design which incorporates business thinking and does it really address the business needs?*

   The state of the art of the service-oriented design is discussed in chapter 2. Among various existing approaches, we selected several works which incorporate business thinking into their design approaches.

   The main features of the selected approaches were compared and the results were shown in the table 2.

2. *What is the business model/ontology underlying the business services?*

   We selected a well-known business modeling ontology - REA as the basis of the proposed service modeling language. As we described in chapter 2, REA and its extensions are strong enough to identify service-resource relationships and the value co-creation. Therefore, we developed a service metamodel to define the construct and their relationships based on REA ontology. The Business Service and Resource Modeling (BSRM) language is defined using the constructs that were defined in the metamodel.

3. *How is the proposed service modeling language represented?*

   A modeling language can be represented textually or graphically. We chose a graphical representation for the BSRM language called BSRM notation. The advantage of using BSRM notation is that the different concepts are easier to recognize because of the different shapes. However, instead of introducing a new modeling notation, it is possible to use other notations, in particular UML diagrams with stereotypes. As UML is a universal language, UML diagrams with stereotypes approach have added advantage. In chapter 2, we introduced the BSRM notation.

4. *How does the proposed framework support the designer building enterprise model to view the business activities in the entire enterprise as services?*

   The proposed service modeling framework provides guided steps to build the enterprise model. The business service patterns which cover the selected activities of manufacturing and service domain facilitate the designer with domain concepts. If patterns are not available in the pattern collection, the new patterns can be derived. The pattern structures specification and pattern structure operations can be used to derive patterns to a particular domain. The isolated patterns do not create a model. The model is a collection of patterns. The pattern composition is not arbitrary. The systematic way of pattern composition is described using business service pattern operators. Finally we proposed the design steps to incorporate patterns to an enterprise model.

   The business activities of a company may include different kinds of resources. For example land, capital, physical resources, human resources, services, informational objects, knowledge, skill, experience etc. The proposed modeling approach supports to design all kinds of the resources which relate with both operational and supporting activities of a company. The intentional resources which are relates with coordination or enhance service help to give better insight to the support business activities

5. *How does the proposed framework support the designer to capture the best practices in the business as services?*

   There are several ways to represent the best practices used in the business. The best practices can be represented by reference models, generic models or patterns. The proposed service design framework provides business service patterns. One advantage of using patterns is the patterns represent the best practices in the business. There are two types of patterns (generic patterns and some domain specific patterns) introduced in this dissertation. Both categories help to represent the best practices.

6. *Is the proposed framework flexible enough to capture new business requirements?*

   It is clear that business service patterns can be used for standardized business activities. But today's business requirements are highly volatile. The proposed business service modeling framework is flexible to capture the new business requirements into the design. The *outsourcing* pattern is one good example. The outsourcing pattern allows merging any external service to existing design. Secondly, the framework provides an opportunity to model extensibility through the *Merge* operator and specialization.

7. *How does the proposed framework support to synchronise the business service model with other models?*

   The business service model integration allows to relate the business service patterns with different architectural layers. We presented two levels of mapping in this research work, horizontal mapping and vertical mapping. The horizontal mapping was discussed in chapter 3 using the BSRM mapping with conceptual data model, e^3-value network. We presented the vertical mapping using the BSRM and process model. Further, pattern-based service integration metamodel, which is in chapter 4, describes the integration of BSRM with PIM and PSM levels. The mapping rules between BSRM and BPMN relates CIM level with PIM. The adaptation of BSPs and the web services through adaptation guidelines relates BSRM patterns with PSM level.

8. *Is the proposed service design framework truly service-oriented?*

   From an architectural perspective, some of the key features of the service-orientation are modularity, loose coupling, reusability and extensibility (Alter, 2012). These features are usually associated with the software services. As we argued in chapter one, the service-oriented thinking at business level is also essential. We analyse the SOA features with respects to business service patterns.

   Modularity:
   One of the most important aspects of SOA is the concept of modularity. The modularity means the ability of identifying separate units. In software engineering, modularity consists of decomposability, composability, understandability, continuity, and protection (Meyer, 1997). In this sub section we explain the first three characteristics with respect to BSRM model. Following top-down approach, decomposability is achieved through Sub-Service specification. The decomposability feature is further explained under the section 4.7.1.2- “merge operator used in decomposition”.

   On the other hand if we take a bottom up approach, pattern composition mechanism demonstrates the feature of composability. The business service pattern operator – “*Merge*” (cf. 4.7.1.2- merge operator used in decomposition) facilitates composition of patterns.

   The third feature, understandability can be motivated as follows. In BSRM the smallest unit of pattern always follows the pattern structure of one of the specification out of five (exchange, conversion, sub-service, coordination or enhance). Hence, the smallest unit of the model is well explained using its parameters, constraints etc.

   Loose coupling
   Coupling refers to the number of dependencies between modules. SOA promotes loose coupling of services. BSRM patterns and models consist of the pattern structure specification. The pattern structure specifications are well defined with its dependencies for a given service. Therefore, loose coupling is promoted by the

business service patterns. The composition of patterns is done through an operator called '"*Merge*". As the merging is done through a common object / group of objects, new patterns don't require additional dependencies except the dependency/dependencies of common object/ group of objects.

Reusability
The proposed framework is a pattern based approach. The patterns are re-useable objects. We use 8 pattern structure specifications, to derive each pattern. We use pattern structure specifications and the patterns to model the cases described in chapter 6. In this dissertation (including appendixes) we present 7 pattern structure specifications, 10 patterns for manufacturing, 8 patterns for supporting activities and 4 patterns for service domain. These pattern structure specifications and the patterns are re-used. From the designer's perspective, he is provided collection of pattern structure specification and the patterns to start the design. Hence, the re-usability of the patterns and specifications reduce the design time and the designer is guided with domain concepts. So the reusability is supported in the proposed framework.

Extensibility
Extensibility is the ability of expanding the model for future growth. The business changes include for instance adding new services or outsourcing. The service modelling framework support adding a new service using "Merge" operator and the "Outsourcing is realised by the "Outsourcing" pattern.

9. *How is the proposed service design framework validated in terms of completeness and correctness?*

   The validation of the proposed framework is presented in chapter 6. We selected three case studies from literature and generate the business service model for them. The proposed framework is able to model these cases within the given case description. The advantages of using the proposed approach and the limitations are discussed. It shows the completeness of the proposed framework. We performed several steps to check the correctness of the proposed BSRM. For example, usage of metamodel, feature comparison and implementing with ConceptBase tool. (These are discussed in chapter3).

7.2 Future Work

In this section we describe the future work and directions of this research. Some of them address the limitations which we listed in chapter 1.

- In the present work, the validation is limited to three case studies in literature. One case is a fictional case about Global Bike Inc., a bike manufacturing company presented by SAP (Magel and Word, 2012). The second one is a real world case selected from literature, about wine production presented by S-Cube (S-Cube, 2009). Both cases relate to manufacturing domain. We selected third case from service industry. The third case is about logistic company. We modelled the three cases using BSRM. Therefore, the future works include the validation of the framework using real world case. Another focus of validation is applying the framework to another domain.

- The next step of future work is extending the BSRM language and the notation. The BSRM language and the notation are sufficient to represent the basic construct of the BSRM language. But it doesn't support to model the business rules. Hence, BSRM language and the notation have to be evolved to the next version to overcome the above deficiencies.

- Extending the business service pattern library is another task which is on the agenda. We developed several basic business service patterns in the manufacturing domain. In the next phase we hope to expand the pattern collection with BSPs of a service domain particularly the banking sector.

- The proposed business service modeling framework is rich enough to develop the enterprise model using patterns, operations and operators. We use ConceptBase to automate the design. This automation is not complete. Therefore, we hope to complete the pattern library in ConceptBase and automate the design process.

Appendix A

This appendix provides the implementation of the metamodel (which is discussed in chapter 3) in ConceptBase tool.

```
{*
This file is governed by the Creative Commons license
   Attribution-NonCommercial 3.0 Unported
   http://creativecommons.org/licenses/by-nc/3.0/
   http://creativecommons.org/licenses/by-nc/3.0/legalcode

Extended licenses, in particular commercial licenses, can be
obtained from the
author of the source code.
Any re-distribution as source code must acknowledge the original
author of this file.
*}

{
*
* File: SERVICE-M2.sml
* Author: Jeewanie Jayasinghe Arachchig, Hans Weigand, Manfred
Jeusfeld
* Creation: 06-Mar-2010 (2-Jan-2013/M.Jeusfeld)
* ----------------------------------------------------------------
* M2 level for the SERVICE metamodel
*
* Copyright (C) 2013 by Jeewanie Jayasinghe Arachchig, Hans
Weigand, Manfred Jeusfeld

*
}

{* ** M2 ** *}

EconomicResourceType in Concept with
  connectedTo
     stockflow: EconomicResourceType;
     flowToService: ServiceType;
     flowToSingleService: ServiceType;
     flowToMultiService: ServiceType
end

ServiceType in Concept isA EconomicResourceType with
  connectedTo
     coordinate: ServiceType;
     flowToResource: EconomicResourceType;
     enhance: ServiceType;
     partOf: ServiceType;
```

```
     take: EconomicResourceType;
     give: EconomicResourceType;
     produce: EconomicResourceType;
     consume:EconomicResourceType;
     use:EconomicResourceType
end


PhysicalResourceType in Concept isA EconomicResourceType end
IndividualizedResourceType in Concept isA PhysicalResourceType
end
NonIndividualizedResourceType in Concept isA
PhysicalResourceType end  {* for example Money *}
MoneyType in Concept isA NonIndividualizedResourceType end

ProperEconomicResourceType in QueryClass isA
EconomicResourceType with
  constraint
     unused: $ not (this in ServiceType) $
end

GoodsType in QueryClass isA EconomicResourceType with
end

IntentionalResourceType in Concept isA EconomicResourceType end

AgentType in Concept with
  connectedTo
    control: EconomicResourceType
end

EconomicResourceType!flowToService isA
EconomicResourceType!stockflow end
ServiceType!flowToResource isA EconomicResourceType!stockflow
end
EconomicResourceType!flowToSingleService isA
EconomicResourceType!flowToService end
EconomicResourceType!flowToMultiService isA
EconomicResourceType!flowToService end


{* FlowServiceTypes are those service types that have at least
one input and one output flow *}
{* This is used to distinguish exchange & conversion services
from enhancing & coordinating  *}
{* services.
*}

FlowServiceType in QueryClass isA ServiceType with
   constraint
     hasInOut: $ exists r1,r2/EconomicResourceType (this
stockflow r1) and (r2 stockflow this) $
end
```

```
{* a sell service is a flow service that generates money from
goods (all non-money resources) *}

SellServiceType in QueryClass isA FlowServiceType with
   constraint
     isSell: $ (forall rout/EconomicResourceType (this stockflow
rout) ==> (rout in MoneyType) ) and
               (forall rin/EconomicResourceType (rin stockflow
this) ==> (rin in GoodsType) )
             $
end

{* a buy service is a flow service that uses money to obtain
goods  *}

BuyServiceType in QueryClass isA FlowServiceType with
   constraint
     exchange: $ (forall rout/EconomicResourceType (this
stockflow rout) ==> (rout in GoodsType) ) and
               (forall rin/EconomicResourceType (rin stockflow
this) ==> (rin in MoneyType) )
               $
end


{* an exchange service is a buy or a sell service *}

ExchangeServiceType in QueryClass isA FlowServiceType with
   constraint
     isexchange: $ (this in SellServiceType) or (this in
BuyServiceType) $
end

{* a conversion service is a flow service where all inputs and
outputs are goods (not money) *}

ConversionServiceType in QueryClass isA FlowServiceType with
   constraint
     allconversion: $ forall r/EconomicResourceType (this
flowToResource r) or (r flowToService this) ==> (r in GoodsType)
$
end


{* a coordination service is simply a service that coordinates
another service *}

CoordinationServiceType in QueryClass isA ServiceType with
   constraint
     csupp: $ exists s/ServiceType (this coordinate s) $
end
```

```
{* an enhancing service is a service that enhances another
service *}

EnhancingServiceType in QueryClass isA ServiceType with
   constraint
     csupp: $ exists s/ServiceType (this enhance s) $
end

{* the rest is called an "other" service *}

OtherServiceType in QueryClass isA ServiceType with
   constraint
     csupp: $ not (this in CoordinationServiceType) and not
(this in EnhancingServiceType) and
              not (this in ConversionServiceType) and not (this
in ExchangeServiceType) $
end


{* atomic services do not have part services *}

AtomicServiceType in QueryClass isA ServiceType with
  constraint
     isAtomic: $ not exists s/ServiceType (s partOf this) $
end


ServiceModelElement in NodeOrLink end

EconomicResourceType isA ServiceModelElement end
EconomicResourceType!stockflow isA ServiceModelElement end
ServiceType!coordinate isA ServiceModelElement end
ServiceType!enhance isA ServiceModelElement end
ServiceType!partOf isA ServiceModelElement end
AgentType isA ServiceModelElement end

ServiceDiagram in Model,Class with   {* synonym for
ComplexActivity *}
  contains
     elem: ServiceModelElement
  rule
     addflow: $ forall a/EconomicResourceType sd/ServiceDiagram
link/EconomicResourceType!stockflow
                        (sd elem a) and From(link,a) ==> (sd
elem link) $;
     addsupport: $ forall a/CoordinationServiceType
sd/ServiceDiagram link/ServiceType!coordinate
                        (sd elem a) and From(link,a) ==> (sd
elem link) $;
     addenhances: $ forall a/EnhancingServiceType
sd/ServiceDiagram link/ServiceType!enhance
```

```
                        (sd elem a) and From(link,a) ==> (sd
elem link) $;
     addpartof: $ forall a/EnhancingServiceType
sd/ServiceDiagram link/ServiceType!partOf
                        (sd elem a) and From(link,a) ==> (sd
elem link) $
end


{* some derived constructs *}

{* resource not used for any service and not refilled *}
UnusedEconomicResourceType in QueryClass isA
ProperEconomicResourceType with
  constraint
     unused: $ not exists st/ServiceType (this flowToService st)
or (st flowToResource this) $
end

{* resource used but not refilled *}
EconomicResourceTypeNotRefilled in QueryClass isA
ProperEconomicResourceType with
  constraint
     unused: $ (exists st1/ServiceType (this flowToService st1))
and
               (not exists st2/ServiceType (st2 flowToResource
this)) $
end

{* resource used only refilled but not used  *}
EconomicResourceTypeOnlyRefilled in QueryClass isA
ProperEconomicResourceType with
  constraint
     unused: $ (not exists st1/ServiceType (this flowToService
st1)) and
               (exists st2/ServiceType (st2 flowToResource
this)) $
end


{* Atomic service types are either
     - enhancing
     - coordinating
     - exchanging money for goods (or vice versa)
     - converting (goods to goods)
   So, this query returns the malformed atomic service types.
*}

UnclassifiableAtomicServiceType in QueryClass isA
AtomicServiceType with
  constraint
     noclass: $ not (this in CoordinationServiceType) and
                not (this in EnhancingServiceType) and
```

```
                not (this in ConversionServiceType) and
                not (this in ExchangeServiceType)
               $
end


{* These rules just map the dedicates relations give/take to the
generic stockflow   *}
{* This alligns the give/take links with the original model
based on stockflow and   *}
{* allows to the GraphViz interface to generate the 'dot' file
of the service models *}

ECArule Take_Map with
  ecarule
        er : $  s/ServiceType r/EconomicResourceType
        ON Tell (s take r)
        IF TRUE
        DO Tell (r flowToService s)
        $
end

ECArule Give_Map with
  ecarule
        er : $   s/ServiceType r/EconomicResourceType
        ON Tell (s give r)
        IF TRUE
        DO Tell (s flowToResource r) $
end

ECArule Produce_Map with
  ecarule
        er : $   s/ServiceType r/EconomicResourceType
        ON Tell (s produce r)
        IF TRUE
        DO Tell (s flowToResource r) $
end

ECArule Consume_Map with
  ecarule
        er : $   s/ServiceType r/EconomicResourceType
        ON Tell (s consume r)
        IF TRUE
        DO Tell (r flowToService s) $
end

ECArule Use_Map with
  ecarule
        er : $   s/ServiceType r/EconomicResourceType
        ON Tell (s use r)
        IF TRUE
        DO Tell (r flowToService s) $
end
```

Appendix B

This appendix provides business service patterns for several operational activities.

Example 1: Creating a new product with assemble and inspection sub-activities

Pattern Name:
BSP- Produce_Intermediate_Stages

Description:
Creating a new product or service always encompasses many intermediate activities such as assembling, inspection, planning etc. The service model for creating a new product which consists of various intermediate stages is described in this example.

Problem:
How do we make a business service model for creating a new product which depicts intermediate steps (assemble and inspection)? We assume that all the intermediate products are consumed to produce the new product.

Assumption:
We assume that the produce has two intermediate steps i.e. assemble and inspection.

Pattern Structure
The service model for creating a new product with intermediate steps is based on Sub-service specification. Each of the intermediate steps follows the Conversion specification (conversion service pattern structure- cf.4, 2.2). All the constraints of sub-service and the conversion pattern structures are to be satisfied when deriving the pattern for creating a new product with intermediate steps.

Solution:
This example includes the intermediate stages of the conversion process such as assembling, inspecting and controlling intermediate product. In other words, the chain of conversion process is splitting the overall conversion process into a chain of smaller conversion processes. We consider the same example in BSP_Produce. But major consideration is splitting the assembling and inspecting into two separate processes. The assembly process creates the assemble product, and the inspection process consumes the assembled product and creates the final product. The waste which is produced during the process is not considered (refer the example of Hruby-8.2). Figure B-2 shows the graphical structure for the produce a product with intermediate stages.

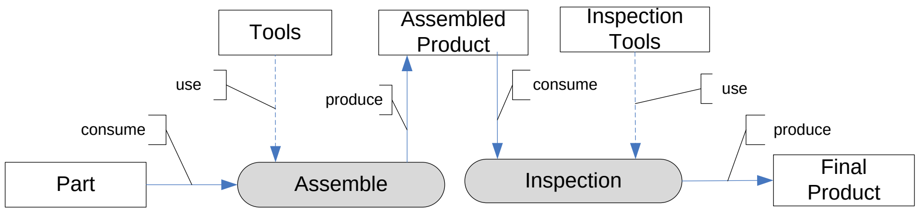

Figure B-1: Business Service Pattern - Produce_Intermediate_Stages

Reference:

- *BSP- Produce_Intermediate_Stages* pattern can be connected with *BSP-Sale* pattern using the *merge* operator. The *FinishedProduct* is the input resource of *BSP-_Cash_Sale*.
- *BSP- Produce_Intermediate_Stages* can be connected with *BSP-Waste* pattern to illustrate the waste management. The *Waste* is not modelled in this pattern. As the waste is a output resource of *Produce* service, it is possible to connect *BSP-Produce* with *BSP-Waste* pattern through *Waste*.

Example 2: Waste Management

Pattern Name:
BSP-Waste

Description:
During the production process, it is common to have waste. The company has to deal with these waste products or materials. There are many ways of handling waste. Eg: dispose, recycling, or getting the service of third party company for waste handling. We consider the simplest way of waste handling in this example, i.e. dispose. Dispose is a conversion process and it is a value decrement activity to the company.

Problem:
How do we make a business service model for dispose waste which is produced in the production process?

Assumptions:
We assume that the dispose of waste as an atomic activity. It uses only the minimal resources.

Pattern Structure
The BSP for disposal of waste is based on Conversion specification. All the constraints of conversion pattern structure are to be satisfied when deriving the pattern. Based on the assumptions, we derive the pattern, by annotating the Conversion structure.

Solution:
Disposal is a conversion service which *uses tools* and *consumes* the *waste*. The *disposal* service adds value to the *waste* (*produce* relationship). Figure B-2 shows the graphical structure for the waste service.

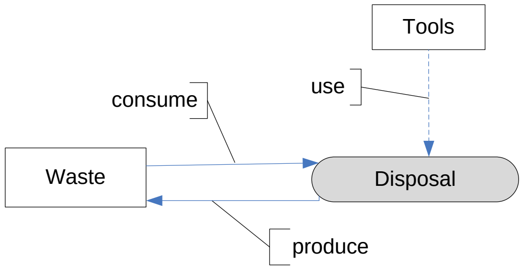

Figure B-2: Business service pattern – disposal of waste

Reference:
- *BSP- Waste* pattern can be connected with *BSP-Produce* pattern using merge operator.
- *BSP- Waste* pattern can be connected with *BSP-Produce_Intermediate_Stages*

Example 3: Product Delivery

Pattern Name:
BSP-Product_Delivery

Description:
Delivering the finished products is one of a main activity of a business. The company can use its own delivery mechanism or use the third party service for the delivery. We consider the simplest way of product delivery and delivery mode can be truck, van etc.

Problem:
How do we make a business service model for product delivery?

Assumptions:
We assume that the delivery as an atomic activity. It uses only the minimal resources.

Pattern Structure
The BSP for delivery of product is based on Conversion specification. All the constraints of conversion pattern structure are to be satisfied when deriving the pattern. Based on the assumptions, we derive the pattern, by annotating the Conversion structure.

Solution:
Product delivery is a conversion service which *uses delivery mode (such as the truck)* and *consumes* the *product*. The *product delivery* service adds value to the *product* by changing its location (*produce* relationship). Figure B-3 shows the graphical structure for the product delivery service.

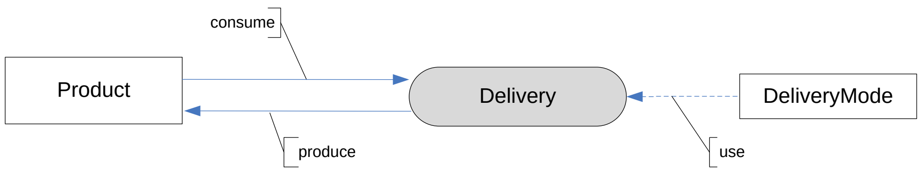

Figure B-3: Business service pattern – product delivery

Reference:
- *BSP-Product_Delivery* pattern can be connected with *BSP-Produce* pattern using merge operator.
- *BSP-Product_Delivery* pattern can be connected with *BSP-Produce_Intermediate_Stages*
- *BSP-Product_Delivery* pattern can be connected with *BSP-Produce_Inventory_Finished_Product*

Example 4: Product Return

Pattern Name:
BSP-Product_Return

Description:
Returning of the sold product is a common activity under certain conditions. The business has to handle the returned product.

Problem:
How do we make a business service model for product return?

Assumptions:
We assume that the product return as an atomic activity. It uses only the minimal resources.

Pattern Structure
The BSP for retuning of product is based on Exchange-Resource specification. All the constraints of Exchange-Resource pattern structure are to be satisfied when deriving the pattern. Based on the assumptions, we derive the pattern, by annotating the Exchange-Resource structure.

Solution:
Product return is an exchange service which *gives money* back to the customer. The company has certain conditions for the retuning of sold products. The product return service *take*s the sold product. Figure B-4 shows the graphical structure for the product return service.

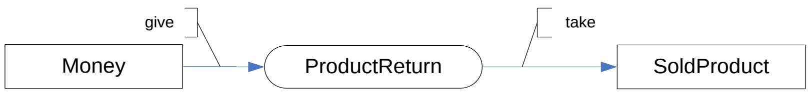

Figure B-4: Business service pattern – product return

Reference:
- *BSP-Product_Return* pattern can be connected with *BSP-Cash_Sale*

Appendix C

Enterprise level business service patterns

In this appendix we demonstrate several enterprise level business service patterns (ESPs). The enterprise service pattern describes the generic activities of the enterprise. The design steps which are discussed in chapter 4, start with a generic Enterprise service pattern.

1. Enterprise service pattern for manufacturing company

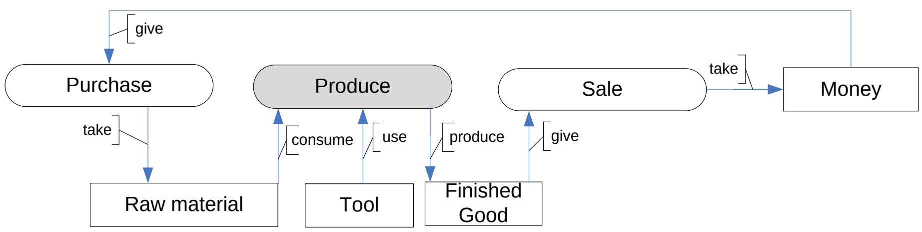

Figure C-1: Enterprise service pattern – manufacturing company

Name:
ESP-Manufacturing

Description:
The primary process of a manufacturing typically refers the transformation of physical inputs into a physical output. The main activities of a manufacturing company include purchasing, produce and sales.

Problem:
How do we make an enterprise service model for a manufacturing company?

Pattern Structure:
Enterprise service pattern for the manufacturing company follows the Exchange-Resource specification and Conversion specification. The first specification is used to model the *Sale* service and the *Purchase* services. The second one is used to model the *Produce* service.

Solution:
Enterprise service pattern for the manufacturing company has three main services namely Sales, Produce and Purchase. The *Sale* service takes *money* and *gives FinishedProduct*. The *Produce* service consumes *RawMaterial* and uses *Tool*. It

produces the *FinishedProduct.* The *Purchase* service gives money and takes *RawMaterial.* Figure: C-1 shows the graphical picture for the ESP-Manufacturing.

2. Enterprise service pattern for trading company

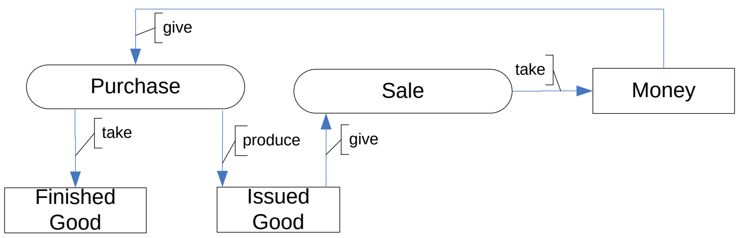

Figure C-2: Enterprise service pattern – manufacturing company

Name:
ESP-Trading

Description:
The primary process of a trading typically refers the exchange of resources or the services. The main activities of a trading company include purchasing and sales.

Problem:
How do we make an enterprise service model for a trading company?

Pattern Structure:
Enterprise service pattern for the trading company follows the Exchange-Resource specification. The specification is used to model the *Sale* service and the *Purchase* services.

Solution:
Enterprise service pattern for the trading company has two main services namely Sales and Purchase. The *Sale* service takes *money* and *gives FinishedProduct*. The *Purchase* service gives money and takes *FinishedProduct.* Figure: C-2 shows the graphical picture for the ESP-Trading.

3. Enterprise service pattern for a Restaurant

Name:
ESP- Restaurant

Description:
The primary process of a restaurant typically refers the sales of services and foods. The main activities of a restaurant include sale of service, generating the service purchasing unprepared food.

Problem:
How do we make an enterprise service model for a restaurant?

Pattern Structure:
Enterprise service pattern for the restaurant follows the Exchange-Service specification, Exchange-Resource specification and the Conversion specification. The first specification is used to model the *Sale* service and the second specification is used to model the purchasing of unprepared food. The third specification is used to model the *RestaurantService*.

Solution:
Enterprise service pattern for the restaurant has two main services namely *Sales* and *RestaurantService*. The *Sale* service takes *money* and *gives RestaurantService*. The *RestaurantService* service consumes unprepared *Food* and uses *Resource*. It produces the *Food.* The *Purchase* service gives money and takes unprepared *Food.* Figure: c-3 shows the graphical picture for the ESP-Restaurant.

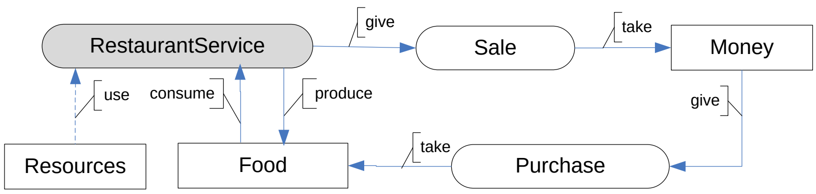

Figure C-3: Enterprise service pattern – Restaurant

References

Alter, S. (2006), The Work System Method: Connecting People, Processes, and IT for Business Results, Larkspur, CA: Work System Press.

Alter, S. (2008), Service system fundamentals: Work system, value chain, and life cycle, International IBM Systems Journal, 47(1), pp.71-86.

Alter, S. (2012), Genuinely Service-Oriented Enterprises: Using Work System Theory to See Beyond the Promise of Efficient Software Architecture, 18th Americas Conference on Information System.

Alexander, C. (1977), A Pattern Language: Towns, Buildings, Construction. USA: Oxford University Press.

Allgaier, M, Heller, M., and Weidner, M. (2010), Towards a Model-based Service Integration Framework for Extensible Enterprise Systems, Enterprise Resource Planning und Transformation von ERP-Systemen 1523, MKWI.

Ambler, T. (2000), Marketing Metrics, Business Strategy Review, 11 (2), pp.59–66.

American Marketing Association (AMA), http://www.marketingpower.com/AboutAMA/Pages/DefinitionofMarketing.aspx, Last accessed on 31/10/2012.

Andersson, B., Bergholtz,M., Grégoire, B., Johannesson, P., Schmitt, M., Zdravkovic,J. (2006), From Business to Process Models – a Chaining Methodology, BUSITAL'06.

Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T., Johannesson, P., Gordijn, J., Grégoire, B., Schmitt, M., Dubois, E., Abels, S., Hahn, A., Wangler, B.,Weigand, H. (2006), Towards a Reference Ontology for Business Models. In: Embley, D.W., Olivé, A., Ram, S. (eds.) ER 2006. LNCS, 4215, pp. 482–496. Springer, Heidelberg.

Andersson, B., Bergholtz, M., Edirisuriya, E., Ilayperuma, T., Johannesson, P. (2005), A declarative foundation of process models. In: Proceedings of the 17th conference on advanced information systems engineering (CAiSE'05). LNCS, 3520. pp. 233–247, Springer, Heidelberg,

Andersson, B., Johannesson, P., Zdravkovic, J. (2009), Aligning Goals and Services Through Goal and Business Modeling, Information systems and e- business modeling, 7(2), pp.143-169.

Arlow. J., Neustadt, I. (2003) , Enterprise patterns and MDA: Building better software with Archetype patterns and UML, Addison Wesley professional.

Arsanjani, A., et al. (2008), SOMA: A method for developing service-oriented solutions. International IBM Systems Journal, 47(3), pp.377-396.

Baader, F., Horrocks, I. & Sattler,U. (2003) ,Description Logics as Ontology Languages for the Semantic Web, LNCS 2605, pp.228-248, Springer.

Baida, Z., Gordijn, J., Saele, H., Omelayenko, B. (2004), A shared service terminology for online service provisioning, ACM International Conference Proceeding Series, 60, pp. 1-10.

Barney, Jay B. (1991), Firm resources and sustained competitive advantage. Journal of Management 17 (March), pp. 99-120.

Becker, J., Kugeler, M., Rosemann, M. (2003), Process Management. Springer.

Bichler, M, Lin K.J. (2006), Service-Oriented Computing: Composition and QoS Issues in Business Services Networks, IEEE Computer, 39(3), pp.99-101.

Bloch, M., Pigneur, Y. and Segev, A. (1996), Leveraging Electronic Commerce for Competitive Advantage: a Business Value Framework, in Proc. of the 9th International EDI-IOS Conference, pp. 91-112.

Borgida, A. (1996), On the relative expressiveness of description logics and predicate logics. Artificial Intelligence. 82(1-2), pp.353-367.

Bottoni, P., Guerra, E. and de Lara, J. (2010), A language-independent and formal approach to pattern-based modelling with support for composition and analysis. Int. Journal Information and Software Technology, 52 (8), pp. 821–844.

Bottoni, P., Guerra, E. and de Lara, J. (2009), Formal Foundation for Pattern-Based Modelling, Proceedings of 12th International Conference on Fundamental Approaches to Software Engineering (FASE), pp.278 – 293.

BPMN OMG (2008), BPMN specification Version 1.1, http://bpmn.org/Documents/BPMN_1-1_Specification.pdf. Last accessed on 31/10/2012.

Brown, A.W. et al. (2005), Realizing service-oriented solutions with the IBM Rational Software Development Platform, IBM Systems Journal, 44(4), pp. 727-752.

Businska, L. and Kirikova, M. (2011), Knowledge Dimension in Business Process Modeling, In Proc. CAiSE Forum, pp.179-187.

Caves, R. E. (1980), Industrial organization, corporate strategy and structure. Journal of Economic Literature, 58, pp. 64-92

Cetina, C. Haugen, O, Zhang, X., Fleurey, F., Pelechano. V. (2009), Strategies for Variability Transformation at Run-time. Proc.13th Int. Software Product Lines Conf (SPLC).

Chesbrough, H. and Spohrer, J. (2006), A Research Manifesto for Services Science, Communications of the ACM, 49 (7), pp. 35-40.

Chesbrough, H. (2011), Open Services Innovation: Rethinking Your Business to Grow and Compete in a New Era, John Wiley & Sons.

Cummins, F.A. (2010), BPM meets SOA. In: Handbook on Business Process Management, vom Brocke , J. and Rosemann, M. (eds), 1, pp. 461-479, Springer, Berlin.

Curdera, F., Khalaf,R., Mukhi, N., Tai, S. , Weerawarna, S. (2003), The next step in services, Communications of ACM 46(10), pp. 29-34.

De Castro, V., Marcos, E., Vara, J.M. (2011), Applying CIM-to-PIM model transformations for the service-oriented development of information systems, Journal of Information and Software Technology, 53 (1) January, pp.87-105.

De Castro,V., Marcos, E., Wieringa, R. (2009), Towards a service-oriented MDA-based approach to the alignment of business processes with IT-systems: from the business model to a web service composition model. In: J. of Cooperative Information Systems 18(2), pp.225-260.

Dieleman, M. (2010), Service identification in practice: Case-study at Van den Bosch Transporten, Master Thesis, Tilburg University.

Dietz, J. (2006), Enterprise Ontology - Theory and Methodology. Springer, Berlin.

Dobson, S., Sterritt, R., Nixon, P. and Hinchey M.(2010), Fulfilling the Vision of Autonomic Computing. IEEE Computer, 43, pp. 35-41.

Douma, S. and Schreuder, H. (2008), Economic Approaches to Organizations (4th Edition) Prentice-Hall.

Eriksson, H. and Penker, M. (2000), Business Modeling with UML – Business Patterns at Work, John-Wiley & Sons, New York.

Erl, Th. (2005), Service-Oriented Architecture: Concepts, Technology, and Design, Upper Saddle River: Prentice Hall PTR.

Erl, Th. (2008), SOA Principles of Service Designing, Prentice Hall.

Fahy, J.Smithee, A. (1999), Strategic Marketing and the Resource Based View of the Firm, Academy of Marketing Science Review (online), 99(10), pp.1-21.

Fensel, D. and Bussler, C. (2002), The Web Service Modeling Framework WSMF. Electronic Commerce Research and Applications, 1(2), pp.113-137.

Fettke, P., Loos, P., Zwicker, J. (2005), Business process reference models: survey and classification, Proceedings of the 3rd international conference on Business Process Management, pp. 469-483, Springer-Verlag Berlin, Heidelberg

Fowler, M. (1996), Analysis Patterns: Reusable Object Models, Addison Wesley Professional.

Frank, U. (2001), Modeling Products for Versatile E-commerce Platforms - Essential Requirements and Generic Design Alternatives. In: Arisawa, H., Kambayashi, Y., Kumar, V., Mayr, H. C., Hunt, I. (eds.): ER 2001 Workshops, HUMACS, DASWIS, ECOMO, and DAMA, pp.444-456, Springer, Berlin.

Froehle, C.M., Roth, A.V. (2007), A Resource-Process Framework of New Service Development, Production and Operations Management, 16(2), pp.169-188.

Gabriel, E. (2006), Value Chain for Services: A new dimension of “Porter’s Value Chain. Int .Journal of IMS

Gailly, F., Laurier, W., Poels, G. (2008), Positioning and Formalizing the REA enterprise ontology, Journal of Information Systems, 22(2), pp. 219-248.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995), Design Patterns: Elements of Reusable Object-Oriented Software, Addison Wesley,

Geerts, G. and McCarthy, W. E. (1999), An Accounting Object Infrastructure For Knowledge-Based Enterprise Models. IEEE Int. Systems & Their Applications, pp. 89-94.

Geerts, G., McCarthy, W. E. (2002), An ontological analysis of the economic primitives of the extended-REA enterprise information architecture, International Journal of Accounting Information Systems 3(1), pp 1-16.

Geerts, G.L., McCarthy, W.E. (2004), The Ontological Foundation of REA Enterprise Information Systems. Michigan State University.

Gigch,J.V. (1991), System design and modeling and metamodeling, Plenum Press New York, NY, USA.

Goedertier, S., Haesen, R. & Vanthienen, J. (2007), EM-BrA2CE v0.1: A vocabulary and execution model for declarative business process modeling. FETEW Research Report KBL-0728, KU Leuven.

Gordijn J, Akkermans J.M., van Vliet J.C. (2000), Business modeling is not process modeling. In: Conceptual modeling for e-business and the web. LNCS, 1921. Springer, Heidelberg

Gordijn, J., Akkermans, J.M. (2001), Designing and Evaluating E-Business Models, IEEE Intelligent Systems, 16, 4, pp.11-17.

Graeber, D. (2001), Toward an anthropological theory of value: The false coin of our own dreams. Palgrave, New York.

Griffioen, P.R., Elsas, P. I. & Van de Riet, R. P. (2000), Analyzing Enterprises: The Value Cycle Approach, In Proc. DEXA 2000, LNCS 1873, pp.685-697, Springer.

Gronroos, C. (2000), Service management and marketing: a customer relationship management approach. West Sussex, UK: Wiley.

Grönroos, C. (2008), Service Logic Revisited: Who Creates Value? And Who Co-Creates? International European Management Review, 20 (4), pp.298-314.

Grossman, G.M. (2005), Outsourcing in a Global Economy, Review of Economic Studies 72, pp.135–159.

Gruber, T.R., (1993), A translation approach to portable ontology specification, Knowledge Acquisition, 5(2):pp.199-220.

Guerra, E. and de Lara, J. (2007), Event-driven grammars: Relating abstract and concrete levels of visual languages. Software and System Modeling, 6(3): pp.317-347.

Gummesson, E. (1995), Relationship marketing: its role in the service economy, In W. J. Glynn, & J. G. Barnes (eds.).

Haesen, R. (2009), Designing Information System Services in Information-Intensive Organisations, PhD thesis. Catholic University Leuven.

Henkel, M., Johannesson, P., Perjons, E., Zdravkovic, J. (2007), Value and Goal Driven Design of E-Services, In proc. of IEEE int. conference on e-Business Engineering.

Hevner, A. R., March, S.T., Park, P., and Ram, S. (2004), Design Science in Information Systems Research, MIS Quarterly, 28(1), pp.75-105.

Hill, T. P. (1977), On Goods and Services, Review of Income and Wealth, 23, pp.315-338.

Holley K., Channabasavaiah, K., and Tuggle, Jr., E.M. (2003), Migrating to a Service-Oriented Architecture, IBM Developer Works, Systems Engineering, pp. 410-424.

Hruby, P. (2006), Model-Driven Design of Software Applications with Business Patterns. Springer.

Hruby, P. (2008), Capitalism in UML, Designing Service-Oriented Applications that Understand Your Business, CSC paper

ISO/IEC (2007), Information technology - Business Operational View - Part 4: Business transaction scenarios - Accounting and economic ontology, ISO 15944-4, http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html, Last accessed on 31/10/2012.

Jayasinghe Arachchige J., Weigand, H., and Jeusfeld, M. (2012), Business Service Modeling for the Service-Oriented Enterprise. International Journal of Information System Modeling and Design, 3(1), pp.1-22.

Jayasinghe Arachchige, J. and Weigand, H. (2012)a, A Pattern based Approach for Business Service Integration, Proc. 6th Int. Workshop on Value Modeling and Business Ontology, VMBO 2012.

Jayasinghe Arachchige, J. and Weigand, H. (2012)b, Business Service Integration Using Pattern Composition. CAiSE Workshops: pp.519-530

Jeusfeld, M.A. (2008): Metamodeling and method engineering with ConceptBase. In Jeusfeld, M.A., Jarke, M., Mylopoulos, J. (eds): Metamodeling for Method Engineering, pp. 89-168. The MIT Press

Jeusfeld, M.A., Jarke, M., & Mylopoulos, J. (2009), Metamodeling for Method Engineering, MIT Press.

Kalakota, R., Robinson, M. (1999), e-Business: Roadmap for Success, Addison-Wesley.

Keller, G., Teufel, T. (1998), SAP R/3 Process Oriented Implementation - Iterative Process Prototyping. Addison-Wesley,

Kirchmer, M. (2009). Reference Models to Empower MPE. In High Performance Through Process Excellence, pp. 85-100.

Kohlborn, T., Korthaus , A., Chan , T., and Rosemann, M. (2009), Identification and analysis of business and software services : A consolidated approach, IEEE Transactions on Service Computing, 2(1), pp. 50-64.

Kohlmann, F. and Alt R. (2009), Business-Oriented Service Design in Practice - Learnings from a Swiss Universal Bank , Proceedings First International Symposium on Services Science ISSS'09, Logos, Berlin.

Kotler, P. and Keller, K. (2006), Marketing Management, 12th Edition, Upper Sanddle River, NJ, Prentice-Hall.

Lane, S. M. (1998), Categories for the Working Mathematician. 2nd Edition. Graduate Texts in Mathematics 5. Springer.

Macey, W. H., Schneider, B., Barbera, K. M., Young S. A. (2009), Employee Engagement: Tools for Analysis, Practice, and Competitive Advantage. West Sussex, UK: Wiley-Blackwell.

Magel, S.R. and Word, J. (2012), Integrated Business process with ERP Systems: Written with support of SAP, John-Wiley & Sons.

Maître, B., Aladjidi, G. (1999), Les business models de la nouvelle économie, Dunod.

Malone, T. W. & Crowston, K. (1994), The interdisciplinary study of coordination. ACM Computing Surveys, 26(1), pp.87-119.

Marshall C. (2000), Enterprise modeling with UML: Designing successful software through Business patterns, Addison Wesley Longman, Inc.

McCarthy, W.E. (1982), The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. The Accounting Review.

Mellor, S.J, Scott, K., Uhl, A. and Weise, D. (2002), Model-Driven Architecture (MDA), OOIS 2002 Workshops, Springer-Verlag Berlin Heidelberg , LNCS 2426, pp. 290-297.

Meyer, B. (1997), Object Oriented Software Construction. Prentice Hall, 1997, pp. 39–48.

Miller, J., Mukerji, J. (2003), MDA Guide Version 1.0.1. Document No.omg/2003-06-01, http://www.omg.com/mda, Last accessed on 31/10/2012

Moscinat, A, Binder, W., Jazayeri, M. (2010), Runtime Adaptability through Automated Model Evolution. Proc. 14th IEEE Enterprise Distributed Object Computing Conference, pp.217-226.

Nayak, N., Nigam, A., Sanz, J., Marston, D., Flaxer, D. (2006), Concepts for Service-Oriented Business Thinking, IEEE International Conference on Services Computing (SCC'06).

Nayak, N., Linehan, M., et al. (2007), Core business architecture for a service-oriented enterprise. IBM Systems Journal, 46(4), pp.723-742.

Nooteboom, B. (2007), Service value chains and effects of scale, Service Business 1;pp.119–139.

OASIS –Reference Model for Service Oriented Architecture 1.0,https://www.oasis-open.org/committees/download.php/16628/wd-soa-rm-pr1.pdf, Last accessed on 31/10/2012.

OMG (2009), http://www.omg.org/spec/SoaML/, Last accessed on 31/10/ 2012.

Organization for Economic corporation and development, http://www.oecd.org/statistics, Last accessed on 31/10/2012.

Osterwalder, A. (2004), The Business Model Ontology, Ph.D. thesis, HEC Lausanne.

Osterwalder, A., Pigneur, Y. (2002), An e-Business Model Ontology for Modeling e-Business, 15th Bled Electronic Commerce Conference e-Reality: Constructing the e-Economy.

Papazoglou, M. & van den Heuvel, W.J. (2006), Service-oriented design and development methodology. International Journal of Web Engineering and Technology, 2(4), pp.412-442.

Papazoglou, M. (2007), Web Services: Principles and Technology. Boston, Pearson Prentice Hall.

Pistore, M., Traverso, P., Paolucci M., Wagner, M. (2009), From Software Services to a Future Internet of Services. In: G. Tselentis et al. (eds), Towards the Future Internet. IOS Press.

Pijpers, V., Gordijn, J. (2007), Bridging business value models and process models in aviation value webs via possession right, In Proceedings of the 39th Hawaii International Conference On System Sciences (HICCS). IEEE.

Porter, M. (1985), Competitive Advantage, Free Press, New York.

Prahalad., C.K. and Krishna, M.S. (2008), New Age of Innovation: Driving Cocreated Value Through Global Networks, McGraw-Hill.

Roman, D. et al. (2005), Web Service Modeling Ontology, http://www.w3.org/Submission/WSMO/, Last accessed on 31/10/2012,

Romney, M., Steinbart, P. (2008), Accounting Information Systems, 11/E Prentice Hall.

Roy, M., Suleiman, B. and Weber, I. (2010), Facilitating Enterprise Service Discovery for Non-Technical Business Users, Proceedings of the international conference on Service-oriented computing, ICSOC'10.

Schaufeli, W. B., Salanova, M. (2007), Work Engagement: An Emerging Psychological Concept and its Implications for Organizations. In Gilliland, S. W. Steiner, D. D. and Skarlicki D. P. (eds.), Managing Social and Ethical Issues in Organizations, Research in Social Issues in Management, 5, pp.135-177, Greenwich, CT: Information Age Publishers.

Scheithauer, G. , Augustin, S. and Wirtz, G. (2009), Business Modeling for Service Engineering: Toward an integrated Procedure Model, SEKE 2009: pp.322-327.

Schmidt, K. & Simone, C. (1996), Coordination mechanisms: Towards a conceptual foundation of cscw systems design. Computer Supported Cooperative Work, 5(2/3), pp.155-200.

Schuster, R. Motal, T. (2009), From e^3-value to REA: Modeling multi-party eBusiness Collaborations, Commerce and Enterprise Computing, CEC '09. IEEE Conference on Congress on Evolutionary Computation, pp.202-208.

S-Cube (2009), http://www.s-cube-network.eu/results/deliverables/wp-ia-2.2/CD-IA-2.2.2_Collection_of_industrial_best_practices_scenarios_and_business_cases.pdf , Last accessed on 31/10/2012.

Siau, K. & Rossi, M. (2008), Evaluation techniques for systems analysis and design modelling methods – a review and comparative analysis. Information Systems Journal, no. doi: 10.1111/j.1365-2575.2007.00255.x.

Simons, R. (2000), Performance Measurement and Control Systems for Implementing Strategy. Prentice Hall.

Stabell, C., Fjeldstad, O. (1998), Configuring Value for Competitive Advantage: On Chains, Shops, and Networks, Strategic Management Journal, 19(5), pp. 413-437.

Staudt, M., Jarke, M., Jeusfeld, M. & Nissen, H. (1993), Query classes. 3rd International Conference on Deductive and Object-Oriented Databases (DOOD'93), Scottsdale, Ariz.

Staudt, M., Nissen, H.W, Jeusfeld, M.A. (1994), Query by class, rule and concept. In Applied Intelligence, Special Issue on Knowledge Base Management, Mylopoulos, J. (eds.), 4, pp.133-156.

The Stencil Group, (2001), Defining Web Services.

Timmers, P., (1998), Business Models for Electronic Markets, Journal on Electronic Markets, 8(2), pp. 3-8.

UML, 2010, http://www.omg.org/spec/UML/, Last accessed on 31/10/2012.

UN/CEFACT Modelling Methodology (UMM) User Guide (2003), http://www.unece.org/cefact/umm/UMM_userguide_220606.pdf, Last accessed on 31/10/2012.

UN/CEFACT (2008), UN/CEFACT's Modeling Methodology (UMM), UMM Meta Model 2.0.

Uschold, M., Gruninger, M. (1996), Ontologies: Principles, Methods, and Applications. Knowledge Engineering Review 11(2), pp.93–155.

Vandenbossche, E.A.P. (2007), A Pattern for Global Payment Optimization via REA Ontology, REA-25 (special track II) Enterprise Model Conference.

Vargo, S. L. & Lusch, R. F. (2004), Evolving To a New Dominant Logic for Marketing, International Journal of Marketing, 68 (January), pp.1-17.

Weigand, H. and Elsas, P. (2012), Model-based auditing using REA, International Journal of Accounting Information Systems, 13(3), pp. 287–310.

Weigand, H., Johannesson, P. , Andersson, B., Bergholtz, M. (2009), Value-Based Service Modeling and Design: Toward a Unified View of Services, Proceedings of the 21st International Conference on Advanced Information.

Weigand, H., Johannesson, P., Andersson, B., Jayasinghe Arachchige, J., and Bergholtz, M. (2011), Management Services – A Framework for Design, CAiSE 2011: pp.582-596.

Weigand, H., Johannesson, P., Andersson, B., Bergholtz, M., Jayasinghe Arachchig, J. (2010), Closing the User-Centric Service Coordination Cycle, CAiSE Forum.

Weigand, H., van den Heuvel, W.J. (1999), Meta-patterns for electronic commerce transactions based on the Formal Language for Business Communication (FLBC), International Journal of Electronic Commerce, 3(2), pp.45-66.

Weigand, H.,Johannesson, P., Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T. (2007) , Value object analysis and the transformation from value model to process model, In Enterprise Interoperability, Springer London, pp.55-65.

Weigand, H.,Johannesson, P., Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T. (2006), On the Notion of Value Object. CAiSE 2006, pp.321-335.

Wenger, E. (1998), Communities of practice. Learning, meaning and identity. London: Cambridge University Press.

Wernerfelt, B. (1984), A resource-based view of the firm. Strategic Management Journal. 5 (April-June), pp.171-180

Wieringa, R.J. and Heerkens, J.M.G. (2006), The methodological soundness of requirements engineering papers: a conceptual framework and two case studies. Requirements engineering, 11 (4). pp. 295-307.

Wieringa, R.J., Pijpers,V., Bodenstaff, L., Gordijn,J. (2008), Value-Driven Coordination Process Design Using Physical Delivery Models, In: 27th International Conference on Conceptual Modeling, pp.216-231, Springer.

Yu, J., Sheng, Q., Swee, J. (2010), Model-Driven Development of Adaptive Service-Based Systems with Aspects and Rules. In: Chen, L., Triantfillou, P., Suel, T. (eds), Proc. WISE 2010, pp.548-563, Springer.

Zdravkovic, J., Ilayperuma, T. (2010), A Model-driven Approach for Designing E-Services Using Business Ontological Frameworks, Enterprise Distributed Object Computing Conference (EDOC), 2010 14th IEEE International, pp.121-130.

Zimmerman, O., Krogdahl, P., Gee, C. (2004), Elements of Service Oriented Analysis and Design, http://www-128.ibm.com/developerworks/library/ws-soad1/ , Last accessed on 31/10/2012.

SIKS Dissertation Series

2009-01 Rasa Jurgelenaite (RUN)
Symmetric Causal Independence Models

2009-02 Willem Robert van Hage (VU)
Evaluating Ontology-Alignment Techniques

2009-03 Hans Stol (UvT)
A Framework for Evidence-based Policy Making Using IT

2009-04 Josephine Nabukenya (RUN)
Improving the Quality of Organisational Policy Making using Collaboration Engineering

2009-05 Sietse Overbeek (RUN)
Bridging Supply and Demand for Knowledge Intensive Tasks - Based on Knowledge, Cognition, and Quality

2009-06 Muhammad Subianto (UU)
Understanding Classification

2009-07 Ronald Poppe (UT)
Discriminative Vision-Based Recovery and Recognition of Human Motion

2009-08 Volker Nannen (VU)
Evolutionary Agent-Based Policy Analysis in Dynamic Environments

2009-09 Benjamin Kanagwa (RUN)
Design, Discovery and Construction of Service-oriented Systems

2009-10 Jan Wielemaker (UVA)
Logic programming for knowledge-intensive interactive applications

2009-11 Alexander Boer (UVA)
Legal Theory, Sources of Law & the Semantic Web

2009-12 Peter Massuthe (TUE, Humboldt-Universitaet zu Berlin)
Operating Guidelines for Services

2009-13 Steven de Jong (UM)
Fairness in Multi-Agent Systems

2009-14 Maksym Korotkiy (VU)
From ontology-enabled services to service-enabled ontologies (making ontologies work in e-science with ONTO-SOA)

2009-15 Rinke Hoekstra (UVA)
Ontology Representation - Design Patterns and Ontologies that Make Sense

2009-16 Fritz Reul (UvT)
New Architectures in Computer Chess

2009-17 Laurens van der Maaten (UvT)
Feature Extraction from Visual Data

2009-18 Fabian Groffen (CWI)
Armada, An Evolving Database System

2009-19 Valentin Robu (CWI)
Modeling Preferences, Strategic Reasoning and Collaboration in Agent-Mediated Electronic Markets

2009-20 Bob van der Vecht (UU)
Adjustable Autonomy: Controling Influences on Decision Making

2009-21 Stijn Vanderlooy (UM)
Ranking and Reliable Classification

2009-22 Pavel Serdyukov (UT)
Search For Expertise: Going beyond direct evidence

2009-23 Peter Hofgesang (VU)
Modelling Web Usage in a Changing Environment

2009-24 Annerieke Heuvelink (VUA)
Cognitive Models for Training Simulations

2009-25 Alex van Ballegooij (CWI)
"RAM: Array Database Management through Relational Mapping"

2009-26 Fernando Koch (UU)
An Agent-Based Model for the Development of Intelligent Mobile Services

2009-27 Christian Glahn (OU)
Contextual Support of social Engagement and Reflection on the Web

2009-28 Sander Evers (UT)
Sensor Data Management with Probabilistic Models

2009-29 Stanislav Pokraev (UT)
Model-Driven Semantic Integration of Service-Oriented Applications

2009-30 Marcin Zukowski (CWI)
Balancing vectorized query execution with bandwidth-optimized storage

2009-31 Sofiya Katrenko (UVA)
A Closer Look at Learning Relations from Text

2009-32 Rik Farenhorst (VU) and Remco de Boer (VU)
Architectural Knowledge Management: Supporting Architects and Auditors

2009-33 Khiet Truong (UT)
How Does Real Affect Affect Affect Recognition In Speech?

2009-34 Inge van de Weerd (UU)
Advancing in Software Product Management: An Incremental Method Engineering Approach

2009-35 Wouter Koelewijn (UL)
Privacy en Politiegegevens; Over geautomatiseerde normatieve informatie-uitwisseling

2009-36 Marco Kalz (OUN)
Placement Support for Learners in Learning Networks

2009-37 Hendrik Drachsler (OUN)
Navigation Support for Learners in Informal Learning Networks

2009-38 Riina Vuorikari (OU)
Tags and self-organisation: a metadata ecology for learning resources in a multilingual context

2009-39 Christian Stahl (TUE, Humboldt-Universitaet zu Berlin)
Service Substitution -- A Behavioral Approach Based on Petri Nets

2009-40 Stephan Raaijmakers (UvT)
Multinomial Language Learning: Investigations into the Geometry of Language

2009-41 Igor Berezhnyy (UvT)
Digital Analysis of Paintings

2009-42 Toine Bogers (UvT)
Recommender Systems for Social Bookmarking

2009-43 Virginia Nunes Leal Franqueira (UT)
Finding Multi-step Attacks in Computer Networks using Heuristic Search and Mobile Ambients

2009-44 Roberto Santana Tapia (UT)
Assessing Business-IT Alignment in Networked Organizations

2009-45 Jilles Vreeken (UU)
Making Pattern Mining Useful

2009-46 Loredana Afanasiev (UvA)
Querying XML: Benchmarks and Recursion

2010-01 Matthijs van Leeuwen (UU)
Patterns that Matter

2010-02 Ingo Wassink (UT)
Work flows in Life Science

2010-03 Joost Geurts (CWI)
A Document Engineering Model and Processing Framework for Multimedia documents

2010-04 Olga Kulyk (UT)
Do You Know What I Know? Situational Awareness of Co-located Teams in Multidisplay Environments

2010-05 Claudia Hauff (UT)
Predicting the Effectiveness of Queries and Retrieval Systems

2010-06 Sander Bakkes (UvT)
Rapid Adaptation of Video Game AI

2010-07 Wim Fikkert (UT)
Gesture interaction at a Distance

2010-08 Krzysztof Siewicz (UL)
Towards an Improved Regulatory Framework of Free Software. Protecting user freedoms in a world of software communities and eGovernments

2010-09 Hugo Kielman (UL)
A Politiele gegevensverwerking en Privacy, Naar een effectieve waarborging

2010-10 Rebecca Ong (UL)
Mobile Communication and Protection of Children

2010-11 Adriaan Ter Mors (TUD)
The world according to MARP: Multi-Agent Route Planning

2010-12 Susan van den Braak (UU)
Sense making software for crime analysis

2010-13 Gianluigi Folino (RUN)
High Performance Data Mining using Bio-inspired techniques

2010-14 Sander van Splunter (VU)
Automated Web Service Reconfiguration

2010-15 Lianne Bodenstaff (UT)
Managing Dependency Relations in Inter-Organizational Models

2010-16 Sicco Verwer (TUD)
Efficient Identification of Timed Automata, theory and practice

2010-17 Spyros Kotoulas (VU)
Scalable Discovery of Networked Resources: Algorithms, Infrastructure, Applications

2010-18 Charlotte Gerritsen (VU)
Caught in the Act: Investigating Crime by Agent-Based Simulation

2010-19 Henriette Cramer (UvA)
People's Responses to Autonomous and Adaptive Systems

2010-20 Ivo Swartjes (UT)
Whose Story Is It Anyway? How Improv Informs Agency and Authorship of Emergent Narrative

2010-21 Harold van Heerde (UT)
Privacy-aware data management by means of data degradation

2010-22 Michiel Hildebrand (CWI)
End-user Support for Access to\\ Heterogeneous Linked Data

2010-23 Bas Steunebrink (UU)
The Logical Structure of Emotions

2010-24 Dmytro Tykhonov
Designing Generic and Efficient Negotiation Strategies

2010-25 Zulfiqar Ali Memon (VU)
Modelling Human-Awareness for Ambient Agents: A Human Mindreading Perspective

2010-26 Ying Zhang (CWI)
XRPC: Efficient Distributed Query Processing on Heterogeneous XQuery Engines

2010-27 bMarten Voulon (UL)
Automatisch contracteren

2010-28 Arne Koopman (UU)
Characteristic Relational Patterns

2010-29 Stratos Idreos(CWI)
Database Cracking: Towards Auto-tuning Database Kernels

2010-30 Marieke van Erp (UvT)
Accessing Natural History - Discoveries in data cleaning, structuring, and retrieval

2010-31 Victor de Boer (UVA)
Ontology Enrichment from Heterogeneous Sources on the Web

2010-32 Marcel Hiel (UvT)
An Adaptive Service Oriented Architecture: Automatically solving Interoperability Problems

2010-33 Robin Aly (UT)
Modeling Representation Uncertainty in Concept-Based Multimedia Retrieval

2010-34 Teduh Dirgahayu (UT)
Interaction Design in Service Compositions

2010-35 Dolf Trieschnigg (UT)
Proof of Concept: Concept-based Biomedical Information Retrieval

2010-36 Jose Janssen (OU)
Paving the Way for Lifelong Learning; Facilitating competence development through a learning path specification

2010-37 Niels Lohmann (TUE)
Correctness of services and their composition

2010-38 Dirk Fahland (TUE)
From Scenarios to components

2010-39 Ghazanfar Farooq Siddiqui (VU)
Integrative modeling of emotions in virtual agents

2010-40 Mark van Assem (VU)
Converting and Integrating Vocabularies for the Semantic Web

2010-41 Guillaume Chaslot (UM)
Monte-Carlo Tree Search

2010-42 Sybren de Kinderen (VU)
Needs-driven service bundling in a multi-supplier setting - the computational e3-service approach

2010-43 Peter van Kranenburg (UU)
A Computational Approach to Content-Based Retrieval of Folk Song Melodies

2010-44 Pieter Bellekens (TUE)
An Approach towards Context-sensitive and User-adapted Access to Heterogeneous Data Sources, Illustrated in the Television Domain

2010-45 Vasilios Andrikopoulos (UvT)
A theory and model for the evolution of software services

2010-46 Vincent Pijpers (VU)
e3alignment: Exploring Inter-Organizational Business-ICT Alignment

2010-47 Chen Li (UT)
Mining Process Model Variants: Challenges, Techniques, Examples

2010-48 Withdrawn

2010-49 Jahn-Takeshi Saito (UM)
Solving difficult game positions

2010-50 Bouke Huurnink (UVA)
Search in Audiovisual Broadcast Archives

2010-51 Alia Khairia Amin (CWI)
Understanding and supporting information seeking tasks in multiple sources

2010-52 Peter-Paul van Maanen (VU)
Adaptive Support for Human-Computer Teams: Exploring the Use of Cognitive Models of Trust and Attention

2010-53 Edgar Meij (UVA)
Combining Concepts and Language Models for Information Access

2011-01 Botond Cseke (RUN)
Variational Algorithms for Bayesian Inference in Latent Gaussian Models

2011-02 Nick Tinnemeier(UU)
Organizing Agent Organizations. Syntax and Operational Semantics of an Organization-Oriented Programming Language

2011-03 Jan Martijn van der Werf (TUE)
Compositional Design and Verification of Component-Based Information Systems

2011-04 Hado van Hasselt (UU)
Insights in Reinforcement Learning; Formal analysis and empirical evaluation of temporal-difference learning algorithms

2011-05 Base van der Raadt (VU)
Enterprise Architecture Coming of Age - Increasing the Performance of an Emerging Discipline.

2011-06 Yiwen Wang (TUE)
Semantically-Enhanced Recommendations in Cultural Heritage

2011-07 Yujia Cao (UT)
Multimodal Information Presentation for High Load Human Computer Interaction

2011-08 Nieske Vergunst (UU)
BDI-based Generation of Robust Task-Oriented Dialogues

2011-09 Tim de Jong (OU)
Contextualised Mobile Media for Learning

2011-10 Bart Bogaert (UvT)
Cloud Content Contention

2011-11 Dhaval Vyas (UT)
Designing for Awareness: An Experience-focused HCI Perspective

2011-12 Carmen Bratosin (TUE)
Grid Architecture for Distributed Process Mining

2011-13 Xiaoyu Mao (UvT)
Airport under Control. Multiagent Scheduling for Airport Ground Handling

2011-14 Milan Lovric (EUR)
Behavioral Finance and Agent-Based Artificial Markets

2011-15 Marijn Koolen (UvA)
The Meaning of Structure: the Value of Link Evidence for Information Retrieval

2011-16 Maarten Schadd (UM)
Selective Search in Games of Different Complexity

2011-17 Jiyin He (UVA)
Exploring Topic Structure: Coherence, Diversity and Relatedness

2011-18 Mark Ponsen (UM)
Strategic Decision-Making in complex games

2011-19 Ellen Rusman (OU)
The Mind ' s Eye on Personal Profiles

2011-20 Qing Gu (VU)
Guiding service-oriented software engineering - A view-based approach

2011-21 Linda Terlouw (TUD)
Modularization and Specification of Service-Oriented Systems

2011-22 Junte Zhang (UVA)
System Evaluation of Archival Description and Access

2011-23 Wouter Weerkamp (UVA)
Finding People and their Utterances in Social Media

2011-24 Herwin van Welbergen (UT)
Behavior Generation for Interpersonal Coordination with Virtual Humans On Specifying, Scheduling and Realizing Multimodal Virtual Human Behavior

2011-25 Syed Waqar ul Qounain Jaffry (VU))
Analysis and Validation of Models for Trust Dynamics

2011-26 Matthijs Aart Pontier (VU)
Virtual Agents for Human Communication - Emotion Regulation and Involvement-Distance Trade-Offs in Embodied Conversational Agents and Robots

2011-27 Aniel Bhulai (VU)
Dynamic website optimization through autonomous management of design patterns

2011-28 Rianne Kaptein(UVA)
Effective Focused Retrieval by Exploiting Query Context and Document Structure

2011-29 Faisal Kamiran (TUE)
Discrimination-aware Classification

2011-30 Egon van den Broek (UT)
Affective Signal Processing (ASP): Unraveling the mystery of emotions

2011-31 Ludo Waltman (EUR)
Computational and Game-Theoretic Approaches for Modeling Bounded Rationality

2011-32 Nees-Jan van Eck (EUR)
Methodological Advances in Bibliometric Mapping of Science

2011-33 Tom van der Weide (UU)
Arguing to Motivate Decisions

2011-34 Paolo Turrini (UU)
Strategic Reasoning in Interdependence: Logical and Game-theoretical Investigations

2011-35 Maaike Harbers (UU)
Explaining Agent Behavior in Virtual Training

2011-36 Erik van der Spek (UU)
Experiments in serious game design: a cognitive approach

2011-37 Adriana Burlutiu (RUN)
Machine Learning for Pairwise Data, Applications for Preference Learning and Supervised Network Inference

2011-38 Nyree Lemmens (UM)
Bee-inspired Distributed Optimization

2011-39 Joost Westra (UU)
Organizing Adaptation using Agents in Serious Games

2011-40 Viktor Clerc (VU)
Architectural Knowledge Management in Global Software Development

2011-41 Luan Ibraimi (UT)
Cryptographically Enforced Distributed Data Access Control

2011-42 Michal Sindlar (UU)
Explaining Behavior through Mental State Attribution

2011-43 Henk van der Schuur (UU)
Process Improvement through Software Operation Knowledge

2011-44 Boris Reuderink (UT)
Robust Brain-Computer Interfaces

2011-45 Herman Stehouwer (UvT)
Statistical Language Models for Alternative Sequence Selection

2011-46 Beibei Hu (TUD)
Towards Contextualized Information Delivery: A Rule-based Architecture for the Domain of Mobile Police Work

2011-47 Azizi Bin Ab Aziz(VU)
Exploring Computational Models for Intelligent Support of Persons with Depression

2011-48 Mark Ter Maat (UT)
Response Selection and Turn-taking for a Sensitive Artificial Listening Agent

2011-49 Andreea Niculescu (UT)
Conversational interfaces for task-oriented spoken dialogues: design aspects influencing interaction quality

2012-01 Terry Kakeeto (UvT)
Relationship Marketing for SMEs in Uganda

2012-02 Muhammad Umair(VU)
Adaptivity, emotion, and Rationality in Human and Ambient Agent Models

2012-03 Adam Vanya (VU)
Supporting Architecture Evolution by Mining Software Repositories

2012-04 Jurriaan Souer (UU)
Development of Content Management System-based Web Applications

2012-05 Marijn Plomp (UU)
Maturing Interorganisational Information Systems

2012-06 Wolfgang Reinhardt (OU)
Awareness Support for Knowledge Workers in Research Networks

2012-07 Rianne van Lambalgen (VU)
When the Going Gets Tough: Exploring Agent-based Models of Human Performance under Demanding Conditions

2012-08 Gerben de Vries (UVA)
Kernel Methods for Vessel Trajectories

2012-09 Ricardo Neisse (UT)
Trust and Privacy Management Support for Context-Aware Service Platforms

2012-10 David Smits (TUE)
Towards a Generic Distributed Adaptive Hypermedia Environment

2012-1J.C.B. Rantham Prabhakara (TUE)
Process Mining in the Large: Preprocessing, Discovery, and Diagnostics

2012-12 Kees van der Sluijs (TUE)
Model Driven Design and Data Integration in Semantic Web Information Systems

2012-13 Suleman Shahid (UvT)
Fun and Face: Exploring non-verbal expressions of emotion during playful interactions

2012-14 Evgeny Knutov(TUE)
Generic Adaptation Framework for Unifying Adaptive Web-based Systems

2012-15 Natalie van der Wal (VU)
Social Agents. Agent-Based Modelling of Integrated Internal and Social Dynamics of Cognitive and Affective Processes.

2012-16 Fiemke Both (VU)
Helping people by understanding them - Ambient Agents supporting task execution and depression treatment

2012-17 Amal Elgammal (UvT)
Towards a Comprehensive Framework for Business Process Compliance

2012-18 Eltjo Poort (VU)
Improving Solution Architecting Practices

2012-19 Helen Schonenberg (TUE)
What's Next? Operational Support for Business Process Execution

2012-20 Ali Bahramisharif (RUN)
Covert Visual Spatial Attention, a Robust Paradigm for Brain-Computer Interfacing

2012-21 Roberto Cornacchia (TUD)
Querying Sparse Matrices for Information Retrieval

2012-22 Thijs Vis (UvT)
Intelligence, politie en veiligheidsdienst: verenigbare grootheden?

2012-23 Christian Muehl (UT)
Toward Affective Brain-Computer Interfaces: Exploring the Neurophysiology of Affect during Human Media Interaction

2012-24 Laurens van der Werff (UT)
Evaluation of Noisy Transcripts for Spoken Document Retrieval

2012-25 Silja Eckartz (UT)
Managing the Business Case Development in Inter-Organizational IT Projects: A Methodology and its Application

2012-26 Emile de Maat (UVA)
Making Sense of Legal Text

2012-27 Hayrettin Gurkok (UT)
Mind the Sheep! User Experience Evaluation & Brain-Computer Interface Games

2012-28 Nancy Pascall (UvT)
Engendering Technology Empowering Women

2012-29 Almer Tigelaar (UT)
Peer-to-Peer Information Retrieval

2012-30 Alina Pommeranz (TUD)
Designing Human-Centered Systems for Reflective Decision Making

2012-31 Emily Bagarukayo (RUN)
A Learning by Construction Approach for Higher Order Cognitive Skills Improvement, Building Capacity and Infrastructure

2012-32 Wietske Visser (TUD)
Qualitative multi-criteria preference representation and reasoning

2012-33 Rory Sie (OUN)
Coalitions in Cooperation Networks (COCOON)

2012-34 Pavol Jancura (RUN)
Evolutionary analysis in PPI networks and applications

2012-35 Evert Haasdijk (VU)
Never Too Old To Learn -- On-line Evolution of Controllers in Swarm- and Modular Robotics

2012-36 Denis Ssebugwawo (RUN)
Analysis and Evaluation of Collaborative Modeling Processes

2012-37 Agnes Nakakawa (RUN)
A Collaboration Process for Enterprise Architecture Creation

2012-38 Selmar Smit (VU)
Parameter Tuning and Scientific Testing in Evolutionary Algorithms

2012-39 Hassan Fatemi (UT)
Risk-aware design of value and coordination networks

2012-40 Agus Gunawan (UvT)
Information Access for SMEs in Indonesia

2012-41 Sebastian Kelle (OU)
Game Design Patterns for Learning

2012-42 Dominique Verpoorten (OU)
Reflection Amplifiers in self-regulated Learning

2012-43 Withdrawn

2012-44 Anna Tordai (VU)
On Combining Alignment Techniques

2012-45 Benedikt Kratz (UvT)
A Model and Language for Business-aware Transactions

2012-46 Simon Carter (UVA)
Exploration and Exploitation of Multilingual Data for Statistical Machine Translation

2012-47 Manos Tsagkias (UVA)
Mining Social Media: Tracking Content and Predicting Behavior

2012-48 Jorn Bakker (TUE)
Handling Abrupt Changes in Evolving Time-series Data

2012-49 Michael Kaisers (UM)
Learning against Learning - Evolutionary dynamics of reinforcement learning algorithms in strategic interactions

2012-50 Steven van Kervel (TUD)
Ontology driven Enterprise Information Systems Engineering

2012-51 Jeroen de Jong (TUD)
Heuristics in Dynamic Sceduling; a practical framework with a case study in elevator dispatching

2013-01 Viorel Milea (EUR)
News Analytics for Financial Decision Support

2013-02 Erietta Liarou (CWI)
MonetDB/DataCell: Leveraging the Column-store Database Technology for Efficient and Scalable Stream Processing

2013-03 Szymon Klarman (VU)
Reasoning with Contexts in Description Logics

2013-04 Chetan Yadati(TUD)
Coordinating autonomous planning and scheduling

2013-05 Dulce Pumareja (UT)
Groupware Requirements Evolutions Patterns

2013-06 Romulo Gonzalves(CWI)
The Data Cyclotron: Juggling Data and Queries for a Data Warehouse Audience

2013-07 Giel van Lankveld (UT)
Quantifying Individual Player Differences

2013-08 Robbert-Jan Merk(VU)
Making enemies: cognitive modeling for opponent agents in fighter pilot simulators

2013-09 Fabio Gori (RUN)
Metagenomic Data Analysis: Computational Methods and Applications

2013-10 Jeewanie Jayasinghe Arachchige(UvT)
A Unified Modeling Framework for Service Design.